

Calhoun: The NPS Institutional Archive
DSpace Repository

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

1980-03

CASTAG a computer assisted interactive naval wargame

Kelley, Kevin John

Monterey, California. Naval Postgraduate School

<http://hdl.handle.net/10945/17542>

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun



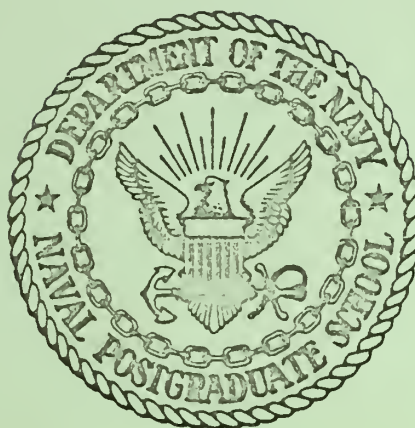
Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

<http://www.nps.edu/library>

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

CASTAG
A COMPUTER ASSISTED INTERACTIVE NAVAL WARGAME

by

Kevin John Kelley

March 1980

Thesis Advisor:

A. Andrus

Approved for public release; distribution unlimited.

Thesis
K27143

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

CASTAG
A COMPUTER ASSISTED INTERACTIVE NAVAL WARGAME

by

Kevin John Kelley

March 1980

Thesis Advisor:

A. Andrus

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) CASTAG A Computer Assisted Interactive Naval Wargame		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; March 1980
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Kevin John Kelley		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE March 1980
		13. NUMBER OF PAGES 157
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Naval Wargame CASTAG SEATAG		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) CASTAG is an interactive two-sided computer wargame which models the movement and detection functions for the manual naval wargame SEATAG, developed at the Naval War College. CASTAG is		

#20 - ABSTRACT - (CONTINUED)

programmed for the CP/CMS time sharing system on the IBM-360
at the Naval Postgraduate School.

Approved for public release; distribution unlimited.

CASTAG
A Computer Assisted Interactive Naval Wargame

by

Kevin John Kelley
Lieutenant, United States Navy
B.A., University of Texas, 1973

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL

March 1980

ABSTRACT

CASTAG is an interactive two-sided computer wargame which models the movement and detection functions for the manual naval wargame SEATAG, developed at the Naval War College. CASTAG is programmed for the CP/CMS time sharing system on the IBM-360 at the Naval Postgraduate School.

TABLE OF CONTENTS

I.	INTRODUCTION AND BACKGROUND -----	9
A.	THE NEED FOR AN INTERACTIVE COMPUTER ASSISTED WARGAME AT THE NAVAL POSTGRADUATE SCHOOL -----	9
B.	APPROACHES TO THE PROBLEM AND LIMITS OF THE THESIS -----	10
C.	SELECTION OF SEATAG -----	10
D.	CHAPTER OUTLINE -----	11
II.	SEATAG, THE MANUAL GAME -----	12
A.	HISTORY AND DEVELOPMENT OF SEATAG -----	12
B.	DESCRIPTION OF THE PLAYING AREA, SCALE, ENVIRONMENT AND PLATFORM CHARACTERISTICS OF SEATAG -----	12
C.	GAME FLOW, AIRCRAFT CARRIER AND SUBMARINE OPERATIONS, AND SURVEILLANCE/INTELLIGENCE PLAY -----	14
D.	DETECTIONS -----	16
E.	PLAY OF ENGAGEMENTS AND CALCULATION OF DAMAGE -----	16
F.	TIME UTILIZATION AND SOME OBSERVATIONS ON ACTUAL MANUAL PLAY -----	17
III.	THE CASTAG PROGRAM -----	18
A.	OVERALL PROGRAM OPERATION -----	18
1.	Overview of CASTAG Inputs -----	18
2.	Dual Terminal Operation -----	20
3.	Description of the CASTAG EXEC Routine ---	20
B.	SELECTION OF THE PROGRAMMING LANGUAGES -----	21
C.	FILE SPACE AND CPU TIME REQUIREMENTS OF THE CASTAG PROGRAM -----	21

D.	CREATION AND ORGANIZATION OF THE DATA BASE	-----	23
1.	Information Required for the Data Base	-----	23
2.	Organization and Format of the Data Base	---	23
E.	PLAYING AREA, TIME AND DISTANCE UNITS	-----	24
F.	DESCRIPTION OF FORTRAN PROGRAM SEARCH	-----	27
1.	MAIN Program	-----	28
2.	Subroutine MOVE	-----	36
3.	Subroutine NUMBER	-----	39
4.	Subroutine DETECT	-----	43
5.	Subroutine SPOOK	-----	52
6.	Subroutine ESM	-----	57
7.	Subroutine DETRAD	-----	62
8.	Subroutine VISUAL	-----	68
9.	Subroutine ASONAR	-----	73
10.	Subroutine NLPCPA	-----	79
11.	Subroutine PSONAR	-----	92
G.	DESCRIPTION OF FORTRAN PROGRAM CHANGE	-----	105
H.	DESCRIPTION OF FORTRAN PROGRAM UPDATE	-----	109
IV.	USER'S GUIDE TO CASTAG	-----	112
A.	CASTAG TERMINAL INPUTS	-----	112
B.	DECODING/ENCODING TYPE NUMBERS	-----	117
C.	CASTAG TERMINAL OUTPUT	-----	121
D.	CASTAG OFFLINE PRINTER OUTPUT	-----	124
1.	Umpire Summary of Detections	-----	125
2.	Red/Blue Cartesian Plot	-----	125
3.	Status of Forces Table	-----	126

4.	Umpires' Cartesian Plot -----	126
5.	Data Base Printout -----	126
E.	PROGRAM FAILURES -----	127
1.	Data Entry Errors -----	127
2.	CP/CMS Failure -----	128
V.	FUTURE DEVELOPMENT AND RESULTS -----	129
A.	IMPROVEMENTS AND EXTENSIONS OF CASTAG -----	129
B.	RESULTS -----	131
	APPENDIX A: ADDITIONAL UNITS -----	132
	APPENDIX B: CHANGE OUTPUT -----	133
	COMPUTER PROGRAM CASTAG EXEC -----	135
	COMPUTER PROGRAM SEARCH -----	137
	COMPUTER PROGRAM CHANGE -----	153
	COMPUTER PROGRAM UPDATE -----	155
	BIBLIOGRAPHY -----	156
	INITIAL DISTRIBUTION LIST -----	157

LIST OF FIGURES

1.	SEATAG Game Flow -----	15
2.	Overall CASTAG Program Flow -----	19
3.	Flowchart of SEARCH MAIN Program -----	35
4.	Flowchart of Subroutine MOVE -----	38
5.	Comparison of Detection Curves -----	40
6.	Flowchart of Subroutine NUMBER -----	42
7.	Flowchart of Subroutine DETECT -----	47
8.	Flowchart of Subroutine SPOOK -----	54
9.	Flowchart of Subroutine ESM -----	59
10.	Flowchart of Subroutine DETRAD -----	64
11.	Flowchart of Subroutine VISUAL -----	71
12.	Flowchart of Subroutine ASONAR -----	76
13.	Calculation of Point (X1,Y1) -----	81
14.	Relative Locations of DRM Endpoints -----	84
15.	Four Quadrant TAU Calculation -----	86
16.	Flowchart of Subroutine NLPCPA -----	89
17.	Comparison of Passive Sonar Detection Curves -----	94
18.	Flowchart of Subroutine PSONAR -----	100
19.	Flowchart of CHANGE Program -----	107
20.	Flowchart of UPDATE Program -----	111

I. INTRODUCTION AND BACKGROUND

A. THE NEED FOR AN INTERACTIVE COMPUTER ASSISTED WARGAME AT THE NAVAL POSTGRADUATE SCHOOL

Several curricula at NPS study wargaming techniques. The courses include manual wargames and computer simulations as part of the course material. At the inception of this study, none of these courses had access to an interactive computer assisted wargame that could handle the mundane and time consuming plotting and computational chores of wargame bookkeeping.

The Warfare Environmental Simulator, WES, is currently being made available at the C³ laboratory at the Naval Postgraduate School. The WES program is run at the Naval Ocean Systems Command (NOSC) in San Diego with outlying command centers such as the C³ Lab and CINCPACFLT in Hawaii served by secure data link. WES games will be played on a strategic global or hemispheric scale and will try to approach real time data handling.

WES has several drawbacks for use in a general unclassified wargaming class at NPS. WES must be played in the C³ laboratory and depends on an outside program located in NOSC's computers. The security requirements and limited use of the C³ laboratory rule out extended use of WES at NPS. In addition, scheduling of WES games to support all curricula would require intercurricular scheduling and concurrence of NOSC.

There is therefore a void in interactive computer assisted wargames available at the Naval Postgraduate School. A medium scale naval wargame, tactical in scope and using the computer for bookkeeping functions is needed to fit the requirements of the wargaming courses. An additional requirement would be for a game to aid training of naval officers in tactical decision making. A game is needed that serves the traditional wargames functions of tactical analysis and education.

B. APPROACHES TO THE PROBLEM AND LIMITS OF THE THESIS

In this thesis the existing manual game SEATAG is partially programmed and implemented on the hardware currently installed at the Naval Postgraduate School's computing facility. The program requires less than two cylinders of disk space and is a self-contained game employing two terminals with appropriate screening of information. Changes in game parameters can be entered during overall program execution. The program calculates movement and processes all detections, with actual engagement results and damage determinations computed manually. The program also provides hard copy output for the participants.

C. SELECTION OF SEATAG

Selection of an existing manual game to convert to an interactive computer assisted program rather than the development of a new game was necessitated by this project's

completion date. SEATAG, a publication of the Naval War College, was selected. SEATAG has the advantage of being an official publication with reasonable and accepted assumptions and methodology. The SEATAG model used for this thesis is unclassified and does not contain classified performance data on the various platforms. SEATAG appealed to the author since it emphasized those items considered important without getting involved in a morass of detail.

Although there was a constant temptation to improve SEATAG in writing this thesis, the computer program is as consistent as possible with the rules, procedures and assumptions of the manual game.

Throughout the remainder of the thesis, SEATAG will refer to the manual game in its second edition from the Naval War College and CASTAG, for Computer Assisted SEATAG, will refer to the game as programmed in this thesis.

D. CHAPTER OUTLINE

Chapter II of this thesis examines SEATAG in terms of the playing area, scale, game flow and procedures for engagements and damage. Chapter III describes CASTAG with descriptions of all CASTAG subprograms and models as well as how interactive two terminal operation was achieved. Chapter IV is a user's guide to the CASTAG program with emphasis on the inputs and outputs. Chapter IV with a copy of the SEATAG rules is sufficient documentation to play CASTAG. Chapter V includes comments on playability and suggestions for additions and improvements to the CASTAG program.

II. SEATAG, THE MANUAL GAME

A. HISTORY AND DEVELOPMENT OF SEATAG

SEATAG, Sea Control Tactical Analysis Game, is currently in its second edition. SEATAG is published by the Center for Advanced Research of the Naval War College.

SEATAG was originally developed in 1975 as a research and analytical tool for evaluating the then new Harpoon anti-shiping missile tactics. The second edition is an improved and revised game which is available in an unclassified version.

B. DESCRIPTION OF THE PLAYING AREA, SCALE, ENVIRONMENT AND PLATFORM CHARACTERISTICS OF SEATAG

SEATAG is a manual naval wargame employing two opposing sides Red, Blue, and an umpire Control Group. SEATAG requires three rooms with sufficient desk space for plotting and record keeping, one room for the Control Group and one for each side, Red and Blue. The minimum number of personnel required to play SEATAG is three.

The Control Group performs the umpire functions of intelligence dissemination, detection and damage disclosures and the analysis of interactions during weapons engagements. A Game Director heads the Control Group.

SEATAG comes complete with several grid sheets and cardboard playing pieces to represent force units to be used as the master plot. The grid square scale may vary from

five to thirty nautical miles per grid square side. Locations are coded by a four digit code representing the center of the square and squares can be subdivided into one hundred sub units.

Capabilities of the various platforms are provided on characteristics in Annex C of the SEATAG Rulebook. Characteristics of forty types of modern United States and Soviet ships and aircraft are supplied. The SEATAG data has been taken from editions of Jane's All the Worlds Fighting Ships and from Combat Fleets of the World 1978/1979: Their Ships, Aircraft, and Armament.

SEATAG is limited to two environmental conditions, "good" weather and "bad" weather. One or both may be used in a game. Bad weather effects aircraft and helicopter operations in the same manner and limits all surface units to 15 knots as well. Bad weather degrades sensor performance in terms of range. In addition, each twenty-four hour day is divided into light, from 0600 to 1800, with the remaining 12 hours dark or night. The only effect of dark is to prohibit visual sightings, and any air operations must be made by IFR (Instrument Flight Rules) capable aircraft.

Rules of engagements with several options are also included in the SEATAG Rulebook.

A supplemental resource variation is included with hypothetical costs for the units. These costs are useful in determining the relative balance of a given scenario.

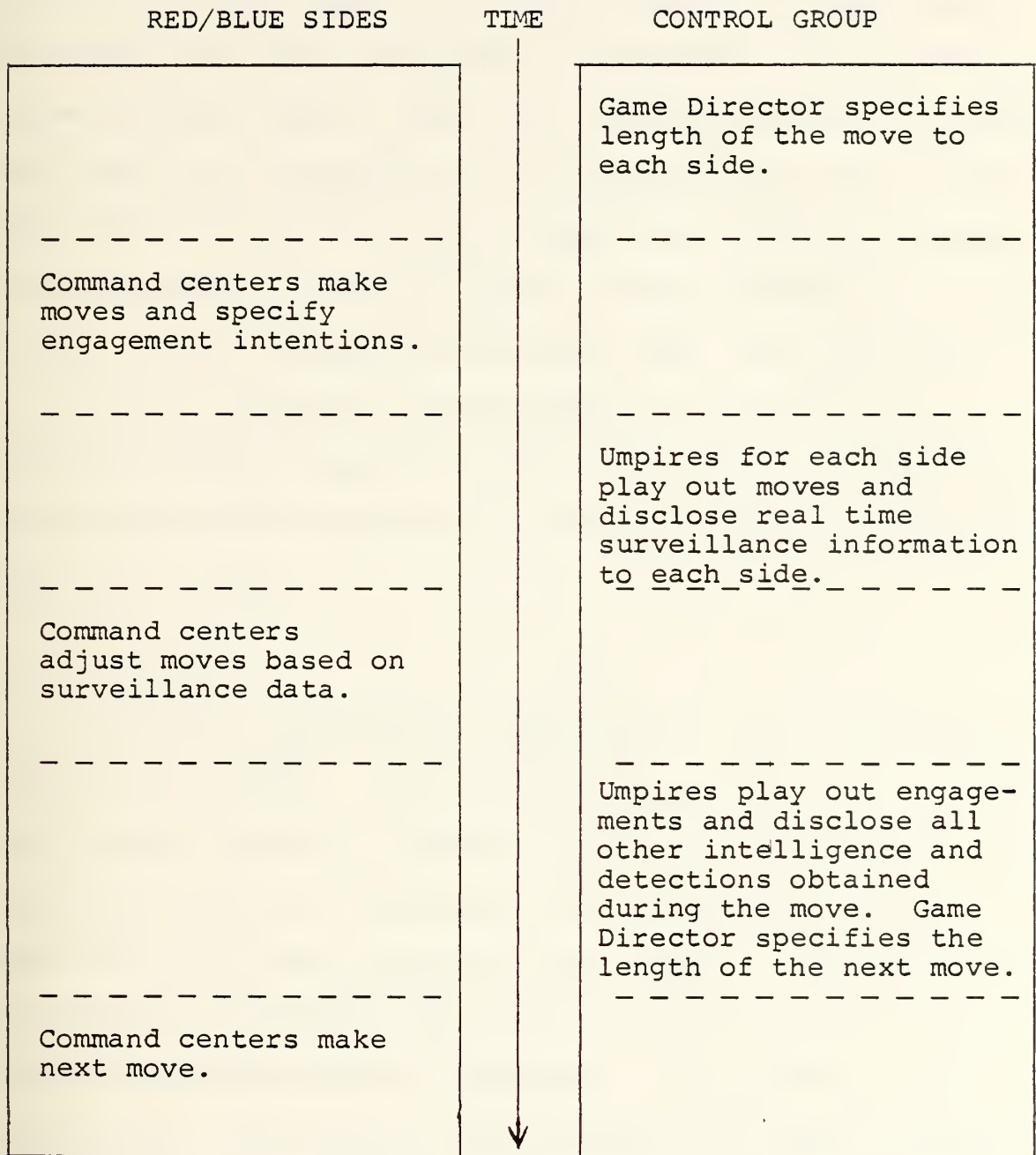
C. GAME FLOW, AIRCRAFT CARRIER AND SUBMARINE OPERATIONS,
AND SURVEILLANCE/INTELLIGENCE PLAY

The game flow is graphically depicted in Figure 1.

The initial decision for each turn is the time duration of that turn. This is specified by the umpires who look ahead to see what the engagement intensity is likely to be. The minimum time step is twenty minutes if the smallest grid size is being used. A one hour minimum is specified for the larger grid scales. Red and Blue then make their moves and specify any engagement intentions to the Control Group who then record the moves and disclose any detections to the appropriate sides. The Red and Blue command centers then adjust moves based on this information. Umpires play out the adjusted moves, play engagements, and provide each side with results, additional detections and the time length of the next move. The adjustment phase allows units to respond to detections as they occur rather than wait until the end of the game move.

The SEATAG rules provide a number of tactical hints on ship positioning, task group organization and submarine warfare. In addition a CV operations section provides a simplified but realistic approach to CV cyclical air operations. An around-the-clock defensive posture is provided along with a selection of surge capabilities against various threats.

Surveillance and intelligence has been distilled into three levels. Each side is assigned or allowed to procure Level I, II, or III. At each day of play at times 0001,



SEATAG GAME FLOW

Figure 1

0601, 1201, and 1801 a random number is drawn for each surface ship, radiating aircraft and submarine traveling in excess of fifteen knots. In addition another random number is drawn each time a unit uses HF equipment. The number drawn in both cases is used with supplied tables to determine the level of information to be provided to the other side. The level ranges from location information only to complete identification, course and speed. Units in EMCON are less likely to have information divulged than those radiating.

Civilian ships and aircraft may be included in the scenario and are operated by the Control Group. Merchants are not allowed to operate in a military capacity except as part of a convoy.

D. DETECTIONS

A detection opportunity occurs when a unit comes within range of any sensor. The information provided when a detection occurs consists of bearing only for ESM detections and complete information including course, speed, altitude or depth and grid posit for most other sensors. Sonobuoys are an exception, with the amount of information determined by the RN generated when the detection was processed. Additional information, like ship or aircraft type and other characteristics, is left totally to the discretion of the umpires.

E. PLAY OF ENGAGEMENTS AND CALCULATION OF DAMAGE

In SEATAG, engagement analysis is segregated into unit versus unit interactions which are treated in a logical order

determined by when they take place. The Control Group is responsible for this function. The Control Group includes a Red and a Blue chief umpire responsible for playing the engagements for their "side", including tactical decisions as required within the framework of specific operation orders or rules of engagement.

Tables and curves are provided to calculate required information such as targeting, weapon expenditures and number of hits inflicted on a target.

F. TIME UTILIZATION AND SOME OBSERVATIONS ON ACTUAL MANUAL PLAY

SEATAG documentation states that 30 to 60 minutes will be required for the Control Group to process each move or turn. Most likely, 80% of that time is spent plotting and doing movement and Closest Point of Approach (CPA) calculations. However, engagement analysis, once the CPA calculation is made, is relatively straightforward and quick.

III. THE CASTAG PROGRAM

A. OVERALL PROGRAM OPERATION

CASTAG consists of the CASTAG EXEC and three FORTRAN programs - SEARCH, CHANGE, and UPDATE. CASTAG EXEC executes the FORTRAN programs in the specified order. SEARCH processes the input, movement, detection and output functions for one game turn. CHANGE provides the user the option to change any unit parameters prior to the next game move. UPDATE combines each terminal's inputs into an updated data base ready for the next game move.

Prior to execution, each terminal must have access to a disk containing a compiled version of the three FORTRAN programs and the CASTAG EXEC program. In addition, one terminal must have access to the data base.

Figure 2 provides an overall program flowchart showing terminal interaction, file management, and input/output.

1. Overview of CASTAG Inputs

CASTAG inputs can be considered in two groups. First is the data base created by the Control Group. The data base is discussed in Section D of this chapter. Second is the terminal inputs required by CASTAG EXEC, SEARCH, and CHANGE. The program functions associated with these inputs are discussed in this chapter under the appropriate program. Chapter IV discusses these same inputs from the point of view of user data entry and as game parameters.

<u>RED TERMINAL</u>	<u>RED EXEC Routine</u>	<u>BLUE EXEC</u>	<u>BLUE TERMINAL</u>
Enter Blue user ID	READ Blue ID	READ Red ID	Enter Red user ID
Enter "T" to transfer data base**	Transfer file FT02F001 to Blue	READ File FT02F001 from Red	Enter "R" to read data base
Enter inputs for program SEARCH	Execute SEARCH which creates file FT02F001	Execute SEARCH which creates file FT02F001	Enter Inputs for program SEARCH
	Offline print file FT02F001	Offline print file FT02F001	
Enter inputs for Program CHANGE	Execute CHANGE	Execute CHANGE	Enter inputs for Program CHANGE
Enter "R" for Red	Transfer file FT03F001 to Blue	Transfer file FT04F001 to Red	Enter "B" for Blue
Enter "Y" when file ready to be read	Read file FT04F001 from Blue	Read File FT03F001 from Blue	Enter "Y" when file ready to be read
	Execute UPDATE Rewind FT03, FT04	Execute UPDATE Rewind FT03, FT04	
Enter "N" to stop, "Y" to play another game turn	READ "Y" or "N"	READ "Y" or "N"	Enter "Y" to play another game turn, "N" to stop
STOP	STOP	STOP	STOP

**Transfer may occur the other way or not at all if both sides have the data base on file.

OVERALL CASTAG PROGRAM FLOW

Figure 2

2. Dual Terminal Operation

Two terminal operation can only be accomplished by each terminal accessing and executing identical programs in tandem. The programs are designated Red or Blue by an input at the terminal. At the NPS computer center this is the only manner in which this kind of a two sided game can be played.

3. Description of the CASTAG EXEC Routine

CASTAG EXEC executes the three compiled FORTRAN programs and manages the file transfers between terminals.

The CASTAG program begins with a one time interactive sequence in which the User Identification number of the other player is input to identify the other terminal for data transfers.

If required, the data base is transferred to the terminal without it.

The program CHANGE is then executed at each terminal by CASTAG EXEC. CHANGE processes changes entered by the user to a unique data file containing the Red or Blue part of the data base at each terminal. Following an entry to identify the terminal as Red or Blue, an interactive sequence at each terminal manages the cross transfer of these two data files so each terminal has the same information. The UPDATE program is then executed resulting in each terminal having a new but identical data base. Finally, the user is given the option to play another turn or stop.

B. SELECTION OF THE PROGRAMMING LANGUAGES

FORTRAN IV (G) was selected as the programming language for CASTAG because it is a widely used and standardized language. The choice of FORTRAN allows the utilization of CASTAG on other machines and systems in locations other than NPS.

The CASTAG EXEC routine is written in the Control Program - 67/Cambridge Monitoring System (CP/CMS) EXECUTIVE language. Implementation of CASTAG on another computer system not using CP/CMS would require reprogramming in that system's file handling language.

C. FILE SPACE AND CPU TIME REQUIREMENTS OF THE CASTAG PROGRAM

CASTAG uses approximately 186k bytes or 233 records of disk storage. This requirement is compatible with the normal two cylinders of disk space assigned to a user at the NPS Computer Center.

Table 1 lists the program and data files and record lengths which make up the CASTAG program. The source code (FORTRAN) versions are listed for reference but are not required to remain on the disk once compiled. They are not included in the total space requirement.

The largest of the three is SEARCH which takes 40 seconds to compile and from 10 to 12 seconds of CPU time to execute. UPDATE and CHANGE run very fast. The time they take is dependent on the number of user inputs.

Table 1

Cylinder Space Requirements for CASTAG

FILE NAME	FILE TYPE	NO. RECORDS	NO. RECORDS
		(20 units/side)	(50 units/side)
CASTAG	EXEC	9	9
SEARCH	FORTTRAN	77	77
SEARCH	TEXT	63	63
UPDATE	FORTTRAN	4	4
UPDATE	TEXT	4	4
CHANGE	FORTTRAN	10	10
CHANGE	TEXT	8	8
FILE	FT01F001	10	125
FILE	FT02F001	5	12
FILE	FT03F001	3	6
FILE	FT04F001	3	6

D. CREATION AND ORGANIZATION OF THE DATA BASE

The data base contains current game time, the total number of units on each side, and all the parameters for every unit in the game. The data base stores the information from turn to turn to provide game continuity.

The data base is created prior to the start of the game using CP/CMS input and edit functions and placed on a data file. This file is created by the player and must be labeled "FILE FT02F001". Once CASTAG execution has begun, the data base normally is never accessed or changed by the user throughout the entire game.

1. Information Required for the Data Base

Game start time, number and type of units per side, each unit's initial position, course, and speed are required from the Control Group or Game Director.

Each unit must have a unique four digit identification number assigned during creation of the data base. In addition, each unit is assigned a second identifying number called a type number. Directions for assigning and decoding these type numbers are located in Chapter IV of the thesis.

The rest of the parameters required for each unit are found in Annex C of the SEATAG rulebook. Characteristics of two additional units are included in Appendix A of the thesis.

2. Organization and Format of the Data Base

The first row always contains the current game time in minutes, the number of Red units and the number of Blue

units. Each of the following rows represents one ship, submarine, or aircraft. All Red units are listed first followed by all Blue units. Units are not required to be in any particular order as regards aircraft, ships, or submarines. Each column in a row contains the parameters for a particular unit. Each column's description is provided in Table 2.

The current game time is read from the data base in format F6.0 by the FORTRAN programs. The number of Red units and the number of Blue units is read in format 2I3. All the rest of the data base is read in by row as 15I4. All entries must be right justified. An integer format was selected to facilitate input and output formatting. Table 3 is an example of how a correctly formatted data base appears on file.

The data base shows 6 Red and 6 Blue units. Game time is 420 minutes.

E. PLAYING AREA, TIME AND DISTANCE UNITS

The game area is the positive quadrant of the cartesian plane. A unit's location is expressed as an (,) co-ordinate to the nearest nautical mile. The maximum X or Y value is 9999. Units with an X or Y value greater than 1000 will not be plotted on the output.

Game time is computed to the nearest minute and it is therefore possible to play game turns of as little as one minute in duration. Speeds are computed to the nearest whole knot.

TABLE 2

Data Base Column Descriptions

ARRAY COLUMN	FIELD	DESCRIPTION
1	1-4	Unit Identification number. Each unit in the game must have a unique 1 to 4 digit unit ID. An ID of 0000 is reserved for units not in play. The unit ID is not used by the program except to check if a unit is in play.
2	5-8	Unit's current speed in knots.
3	9-12	Altitude or depth of the unit. Altitude for aircraft is in thousands of feet. Surface units must be at 0. Submarines may be deep (2), shallow (1), or surfaced (0).
4	13-16	Unit's course in degrees true.
5	17-20	X coordinate of the unit in m.m.
6	21-24	Y coordinate of the unit in n.m.
7	25-28	Type identification number. Each digit represents coded information which is tabulated in Table of Chapter IV.
8	29-32	Unit's maximum speed in knots.
9	33-36	Surface search radar maximum range in n.m.
10	37-40	Air search radar maximum range in n.m.
11	41-44	Sonar maximum range in n.m.
12	45-48	Status of surface search radar. 1 = On, 0 = Off.
13	49-52	Status of air search radar. 1 = On, 0 = Off.
14	53-56	Status of sonar. 1 = On, 0 = Off.
15	57-60	Status of HF communication equipment. 1 = On, 0 = Off.

Table 3
Sample Data Base

420.	6	6												
50	30	0	160	563	4511332	34	25	150	10	1	1	1	0	
51	10	0	160	545	4881333	35	20	150	10	0	0	0	0	
52	10	0	160	547	4481346	34	20	150	10	0	0	0	0	
60	7	1	100	546	4701110	30	100	0	30	0	0	0	0	
0	0	20	90	900	5001281	450	200	0	0	1	0	0	1	
71	10	1	225	510	5101293	120	30	30	2	1	0	1	1	
1011	10	1	95	550	4732110	30	10	0	30	0	0	0	0	
1022	2	2	90	537	4762111	34	10	0	30	0	0	0	0	
333	11	0	90	495	4802333	30	20	150	5	0	0	1	0	
444	12	0	90	510	4762321	33	25	150	5	1	1	0	1	
555	17	0	90	521	4912347	27	20	150	14	1	1	0	0	
666	30	0	90	537	4612347	27	20	150	14	1	1	1	0	

All distances in CASTAG are expressed in nautical miles (n.m.) and all speeds in knots.

F. DESCRIPTION OF FORTRAN PROGRAM SEARCH

SEARCH models the movement and detection elements for one game turn of the manual game. SEARCH is the first program executed by CASTAG EXEC.

SEARCH consists of a main program and ten subroutines.

These are:

1. Subroutine MOVE (TIME)
2. Subroutine NUMBER
3. Subroutine DETECT (TIME)
4. Subroutine SPOOK
5. Subroutine ESM (I)
6. Subroutine DETRAD (I)
7. Subroutine ASONAR (I)
8. Subroutine NLPCPA (I, TIME)
9. Subroutine VISUAL (I, ITYPE)
10. Subroutine PSONAR (I, IPLAT)

The main program will be described first followed by the ten subroutines which make up SEARCH. Each of the eleven sections includes a list of local variables used and a plain language flowchart for that subroutine.

SEARCH uses two computer library routines, UTPLOT and LLRANDOM. UTPLOT is a utility plotting routine available in the SSPLIB library at the NPS Computer Center. This routine will plot more than one set of points on a graph with a different symbol for each call.

LLRANDOM is used to deliver arrays of single precision $U(0,1)$ distributed pseudo random numbers at various locations in SEARCH. LLRANDOM is accessed by CALL RANDOM.

SEARCH uses an extensive common block, which is identical for all the subroutines. Many variables and arrays in common are used repeatedly throughout the program. Some are used in every subroutine. Definitions of the variables in the common block are found in Table 4. The arrays, all of which are in the common block, are defined in Table 5. Referring back to these two tables will be necessary as the information is not repeated in the subroutine description.

1. MAIN Program

The MAIN program dimensions all arrays, initializes variables, zeros out arrays, reads the data base from the file, reads the user's terminal inputs, and either calls subroutine NUMBER or reads the random number file. Following the call to subroutine MOVE, the data base with the new X,Y locations for each unit is read back onto the original file and two new files are created, each with only Red or Blue data. Next, a call to subroutine DETECT calculates all detections for the game turn. Finally, UTPLOT is utilized to output two plots on the offline printer.

The MAIN program is executed by CASTAG EXEC. MAIN calls three subroutines directly: NUMBER, MOVE, and DETECT. This turn's game time is supplied to MOVE and DETECT as an argument. Library routine UTPLOT is called as follows:

```
CALL UTPLOT (XPLOT, YPLOT, NUNIT, R, 1, NGRAPH)
```


Table 4

Description of Variables in the SEARCH Common Block

DTIME	Current game turn's length in minutes
EPS	Used to check for zero in the denominator prior to division. Set to .001.
ID	Tells program whether it is Red of Blue. Also used as third subscript to access the friendly information in array DATA.
IFOE	Third subscript used to access opponents information in array DATA. Set opposite of ID.
ISEED1	Random number generator seed used to determine permanent percentage of maximum detection range.
ISEED 2	Random number seed for numbers generated determining HFDF detections.
ISEED 3	Random number seed for numbers generated to determine intelligence detections.
NRED	Number of Red units. Read from the data base.
NBLUE	Number of Blue units. Read from the data base.
NCOL	Maximum number of columns allowed to be used for information storage in the data base and in array DATA. Set to 15. Maximum allowed by array dimension is 20.
NMAX	Maximum number of units that will have random numbers generated for them. Set to 20. Maximum allowed by array dimension is 50.
NWRITE	Used to designate the off-line printer as the output device for a FORTRAN WRITE statement. Set to 6.
ZTIME	Current culmulative game time in minutes. First number read from data base.

Table 5

DEFINITIONS OF ARRAYS IN SEARCH COMMON BLOCK

CPA (50, 2)	Read array holding calculated ranges in column 1 and bearings in column 2 of CPA's for up to 50 units
DATA (50, 20, 2)	Integer array in which the 2nd and subsequent lines of the data base are stored in. Each row holds the information for one unit, usually subscripted by I. The Jth column holds the Jth parameter for the Ith unit. The columns have the same meaning as in the data base. The K = 1 level is all the Red units' data. The K = 2 level is all the Blue units' data. Note that only 15 of the 20 available spaces for unit data are used.
DICE (50)	Real array used to hold up to fifty U(0,1) random numbers from a call to LLRANDOM.
ESMDET (100, 5)	Integer array holding ESM detection data. Each row holds all the information for a single detection. Columns one through five are interpreted as follows:

<u>Column</u>	<u>Info</u>
1	Type of radar detected
2	Bearing of that radar at CPA
3	2nd and 3rd digits of type number of the detected unit
4	Unit ID of emitting unit
5	Range of CPA of detected radar

HFDF (50, 4)	Integer array holding HFDF detection data. Each holds all the information for a single detection of a unit. Columns one through four are interpreted as follows:
--------------	--

<u>Column</u>	<u>Info</u>
1	Unit ID of detected unit
2	Type number of detected unit
3	X coordinate of detected unit
4	Y coordinate of detected unit

Table 5 (Continued)

INTEL (50, 6)

Integer array holding data revealed during an intelligence segment. Each row contains all the information on a single detected unit. Columns one through six are interpreted as follows:

<u>Column</u>	<u>Info</u>
1	Unit ID of detected unit
2	First two digits of type number of detected unit
3	Second two digits of type number of detected unit
4	Course of unit detected to nearest 10 degrees
5	X coordinate of unit detected
6	Y coordinate of unit detected

MINDEX (10)

Integer array. First nine locations not used. MINDEX (10) holds the total number of detected units that will be plotted.

NROW (2)

Integer array holding number of units on each side. Set to NROW (1) = NRED, NROW (2) = NBLUE

PRMAX (50, 50, 5)

Real array holding the calculated permanent percentage of maximum detection range for each sensor of each unit against every enemy unit. With (I,J,K) as the subscripts, the number represents the Kth sensor on the Ith unit trying to detect the Jth enemy unit. The Kth sensor are:

<u>K</u>	<u>Sensor</u>
1	Surface search radar
2	Air search radar
3	ESM
4	Active sonar
5	Passive sonar and passive sonar ESM

R (4)

Real array holding the grid limits of the X and Y values for the plotting routines.

R(1) = 1000.0 Maximum X
 R(2) = 0.0 Minimum X
 R(3) = 1000.0 Maximum X
 R(4) = 0.0 Minimum Y

Table 5 (Continued)

RADAR (50, 20) Integer array holding all radar and visual detection data. The first ten columns are used for radar data, the second ten for visual data. Each half row represents the detection of a unit. The columns are interpreted as follows:

<u>Column</u>	<u>Radar</u>	<u>Column</u>	<u>Visual</u>
1	Unit ID	11	Unit type number
2	Unit type number	12	Unit altitude
3	Radar type (air/surf)	13	Course in deg T
4	Course (T)	14	X coord
5	X coord	15	Y coord
6	Y coord	16	Bearing at CPA
7	Speed	17	Composition
8	Target composi- tion	18	Range at CPA
9	Range at CPA	19	Unit ID
10	Bearing at CPA	20	Not used

SONAR (50, 20) Integer array holding active sonar, passive sonar, and sonar ESM detection data. The first ten columns are used for active detections and the second ten for passive and ESM detections. Each half row represents the detection of a unit. Columns used as follows:

<u>Column</u>	<u>Active</u>	<u>Column</u>	<u>Passive</u>
1	Unit ID	11	Unit ID
2	Unit type	12	Target type number
3	Unit depth	13	Unit depth
4	Unit course	14	Unit course
5	X coord	15	X coord
6	Y coord	16	Y coord
7	Unit speed	17	Unit speed
8	Range at CPA	18	Range at CPA
9	Bearing at CPA	19	Bearing at CPA
10	Not used	20	Not used

XPLOT (50) Real arrays holding the X coordinates of units to be plotted by UTPLOT.

YPLOT (50) Real arrays holding the Y coordinates of units to be plotted by UTPLOT.

XPLOT and YPLOT are explained in Table 5. R and NUNIT are explained in the local variable definitions. The fifth parameter, 1, tells the routine to plot every (X,Y) point. NGRAPH can equal 1, 2, or 3 indicating if this is the first (1), last (3), or an intermediate (2) set of points to be plotted on a graph.

MAIN reads all the data from disk files required by SEARCH except for three user supplied terminal inputs. These are a random number seed, the length in minutes of the current game turn, and whether the terminal is Red or Blue. These are discussed in Chapter IV. However, the turn length and game time control some additional program functions not apparent to the user.

The program expects to begin the first turn at time 0000. In order to generate the initial PRMAX array either ZTIME must be zero or the turn length (TIME) must be zero. Consequently, to start at an initial game time other than 0000, the length of the first turn must be input as zero. Note that in this case no movement or detections are carried out; only the random numbers are generated by NUMBER. Any time a game turn of zero minutes is played, the only effect is to regenerate the PRMAX array with different values.

Table 6 lists the local variables in this routine. Figure 3 is a flowchart of this routine.

Table 6

MAIN Program, Local Variables

DISK Designate which of four files will input or output
for a READ or WRITE statement as defined below:

<u>Variable</u>	<u>Value</u>	<u>File Type</u>	<u>Purpose</u>
IDISK	1	FT01F001	Stores DRMAX array
NDISK	2	FT02F001	Stores ZTIME, NRED, NBLUE and DATA array
RDISK	3	FT03F001	Stores Red part of DATA array
BDISK	4	FT04F001	Stores Blue part of DATA array

IMIN Number of minutes into latest hour played thus far.

KSEED Random number seed entered by user.

KTIME Calculated number of hours played thus far.

M Number of opposing units detected that are to be
plotted on the first plot.

NGRAPH Parameter for plotting routine designating number
of sets of points on that particular plot.

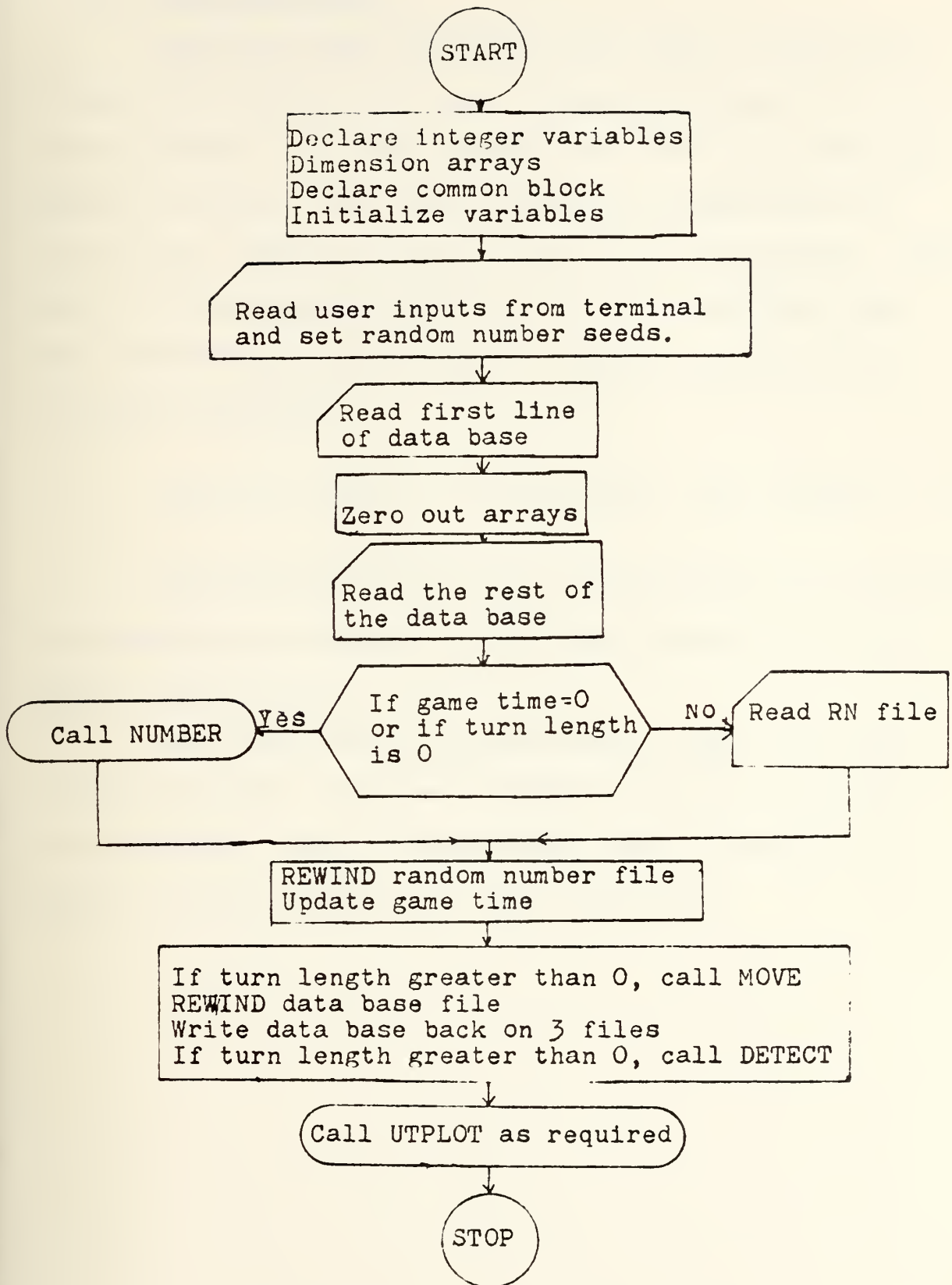
NREAD Input device for FORTRAN READ statement. Set to 5.

NSHIP Maximum number of units arrays can hold. Set to 50.

NTERM Output device for FORTRAN WRITE statement. Set to 5.

NUNIT Number of opposing units to be plotted on second plot.

TIME Length of current turn in minutes; entered by user.



Flowchart of SEARCH MAIN Program

Figure 3

2. Subroutine MOVE

Subroutine MOVE calculates new X and Y coordinates for each Red and Blue unit as a function of course and distance traveled. If a unit crosses the X or Y axis from the positive quadrant, the Y or X coordinate respectively is set to zero, keeping the unit in the positive quadrant. Equation (1) shows the calculation of a new X coordinate; the new Y coordinate is calculated in a similar manner using the cosine function.

$$\text{New X} = \text{Old X} + (.5 + \text{Distance} * \sin(\text{course})) \quad (1)$$

One-half is added so that when the value is truncated for storage as an integer, it in effect rounds off to the nearest nautical mile.

Subroutine MOVE is called once per game turn from the MAIN program. The turn's length in minutes is passed as an argument. Table 7 defines the local variables and Figure 4 is the flowchart for this subroutine.

Table 7

Subroutine MOVE, Local Variables

THOUR	Length of the turn in hours.
RAD	Converts degrees to radians. Set to $\pi/180$.
DIST	Distance the Ith unit moved this turn.
THETA-	Course of the Ith unit converted to radians.
I	Subscripts the row (unit) in array DATA.
K	Subscripts array DATA designating Red or Blue.

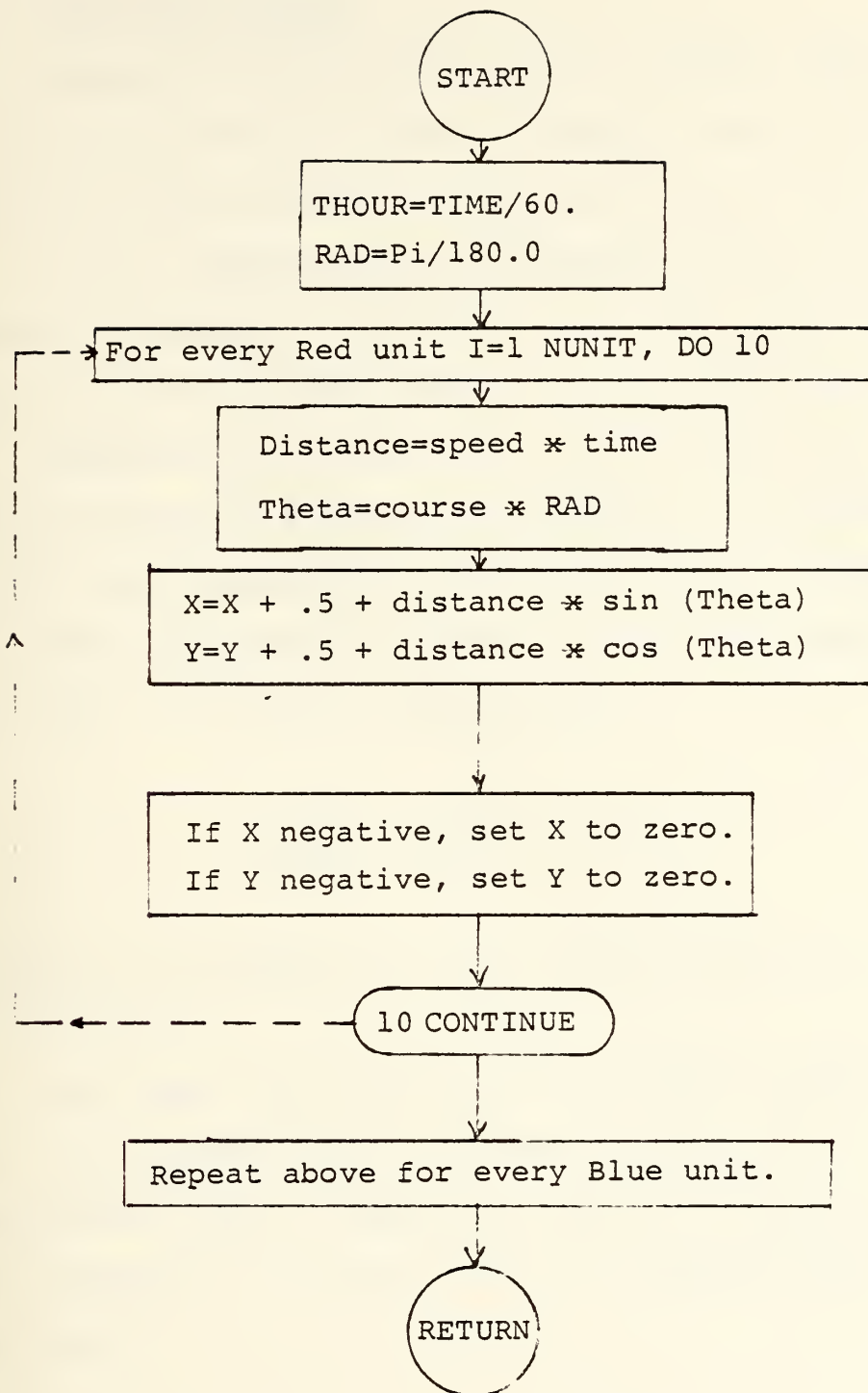


Figure 4
Flowchart of Subroutine MOVE (TIME)

3. Subroutine NUMBER

Subroutine NUMBER draws a U(0,1) random number and calculates the percent of maximum range at which a detection can occur for every sensor on every unit versus every target unit. A U(0,1) random number is drawn for every possibility, regardless of whether a particular unit has a particular sensor or can ever be detected by a particular sensor.

SEATAG uses an empirically derived curve to manually transform the random number into a percent of maximum range. This curve was approximated by the hyperbola in Equation (1), where X equals the percent of maximum detection range (PRMAX) and Y is a U(0,1) random number.

$$\frac{X^2}{(.9)^2} + \frac{Y^2}{(.9)^2} = 1.0 \quad (1)$$

The hyperbola is graphed with the original curve for comparison in Figure 5.

Note that the X and Y axes in Figure 5 have different scales. The maximum error, determined graphically, is 2% of the maximum range. Subroutine NUMBER uses Equation (2) which is Equation (1) solved for X to calculate the actual PRMAX values.

$$X = .81 - Y^2 \quad (2)$$

Subroutine NUMBER is usually called by MAIN only on the first game turn. Thereafter the calculated values are stored from turn to turn on a file.

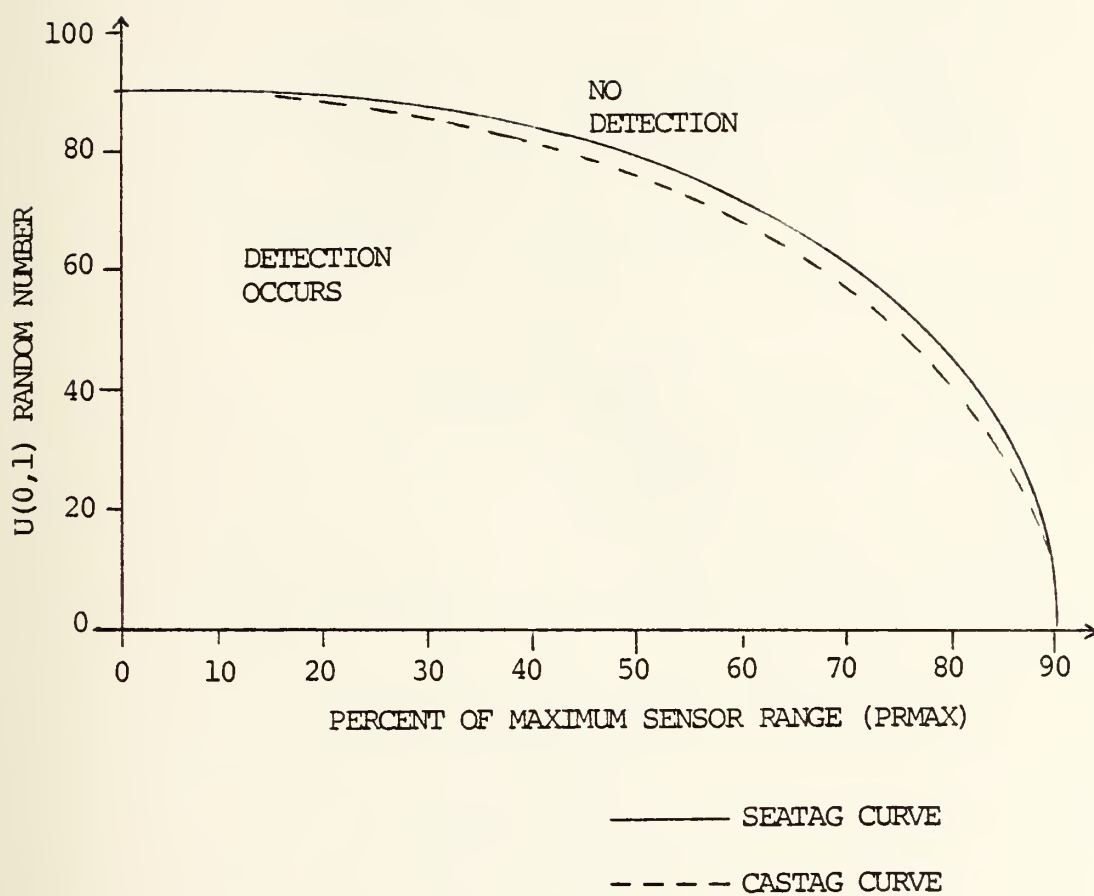


Figure 5
COMPARISON OF DETECTION CURVES

With the exceptions of subscripts I, J, and K, no local variables are used in this subroutine. Figure 6 is the flowchart for subroutine NUMBER.

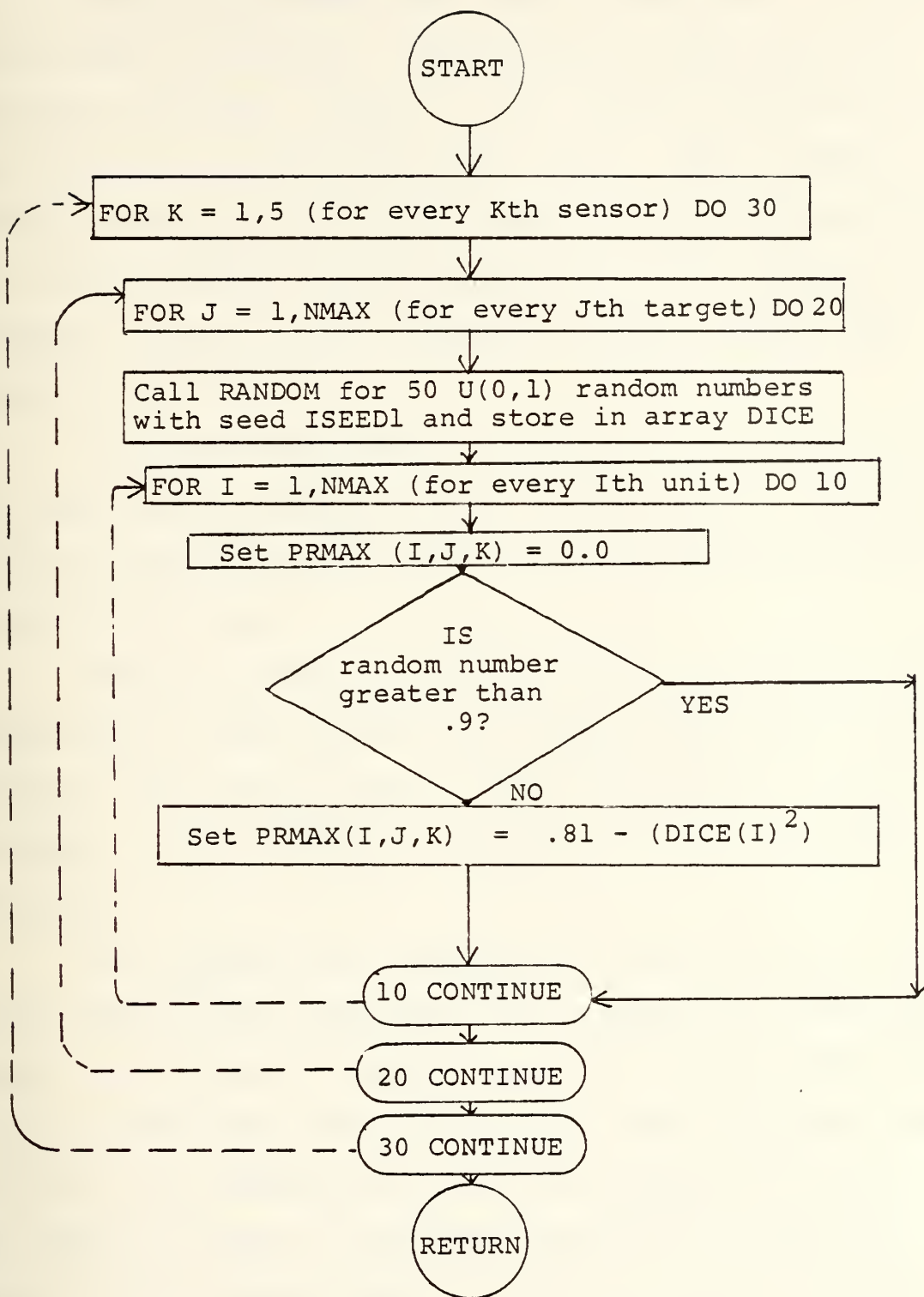


Figure 6

Flowchart for Subroutine NUMBER

4. Subroutine DETECT

The most important and complex subroutine in SEARCH is the DETECT subroutine. The subroutine comprises three functional sections. First, it must decide if the current turn includes a surveillance opportunity. Next, intelligence from enemy units using HF communication is determined. Last, every eligible friendly unit is given the opportunity to detect eligible targets subject to sensor, speed, and platform limitations.

For HFDF intelligence purposes, the program assumes that each side has procured level I intelligence. Accordingly, the only information provided is the grid position of the enemy unit detected.

DETECT performs the umpire function of deciding which sensors may be used against which targets under what circumstances. Some ground rules for this are implicit in the flowchart and some explicitly explained in the following paragraphs.

The program gives submarines at shallow depth some of the capabilities associated with a submarine at periscope depth. In particular, shallow submarines may make ESM and surface search (S/S) radar detections. This represents the ability of the submarine to extend its ESM and S/S radar mast above the water in the same manner as the periscope. Except for ESM against radar emissions, a shallow submarine is still undetectable by means other than sonar. The program does not allow for visual detection by periscope due to the limited maximum range at which such a detection could occur.

Fighter, strike (attack), interceptors and ECM aircraft may only make visual detections. In general radars on board these aircraft are fire control radars without a 360 degree search capability. Their ESM equipment is primarily short range threat warning oriented and is used to provide a warning of fire control or missile homing acquisition or lock on. Also these aircraft are never detectable by ESM in the program due to the narrow sector of the emitters.

Arbitrary limits of ten knots maximum speed and 1000 feet or less altitude were imposed on the SH-3 and KA-25 helicopters before they may attempt a dipping sonar detection.

Subroutine DETECT is called once per turn from the MAIN program. Subroutine DETECT calls the following subroutines.

1. SPOOK
2. ESM (I)
3. DETRAD (I)
4. ASONAR (I)
5. NLPCPA (I, TIME)
6. VISUAL (I, ITYPE)
7. PSONAR (I, IPLAT)

The I argument passed to all the subroutines called except SPOOK designates the row subscript of the unit searching with that particular sensor.

The second argument passed to NLPCPA is the same length of the current turn passed to DETECT. The second

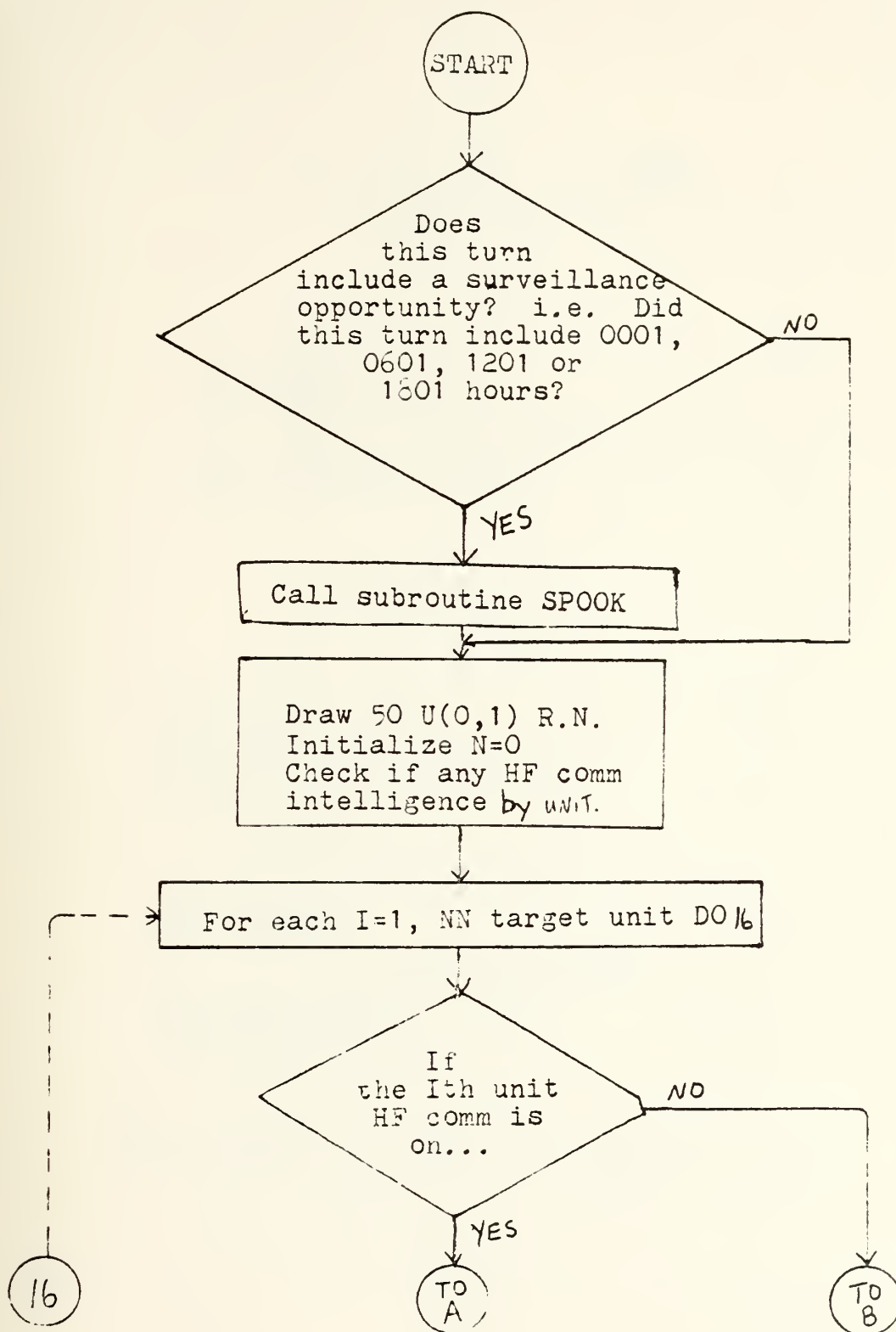
argument passed to VISUAL and PSONAR indicates what kind of unit the Ith unit is.

Table 8 contains the local variables used in DETECT and Figure 7 is a flowchart for DETECT.

Table 8

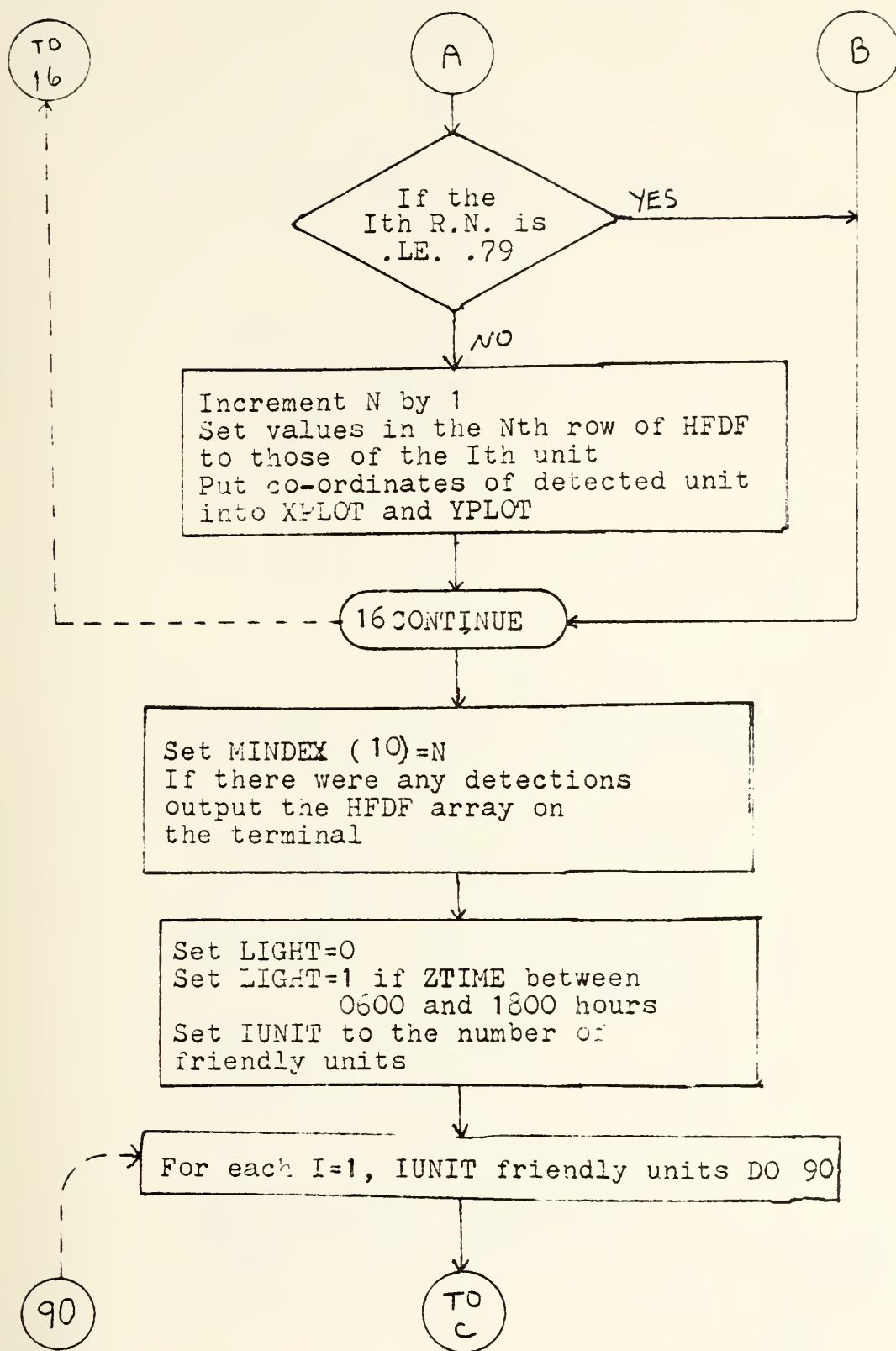
Subroutine DETECT, Local variables

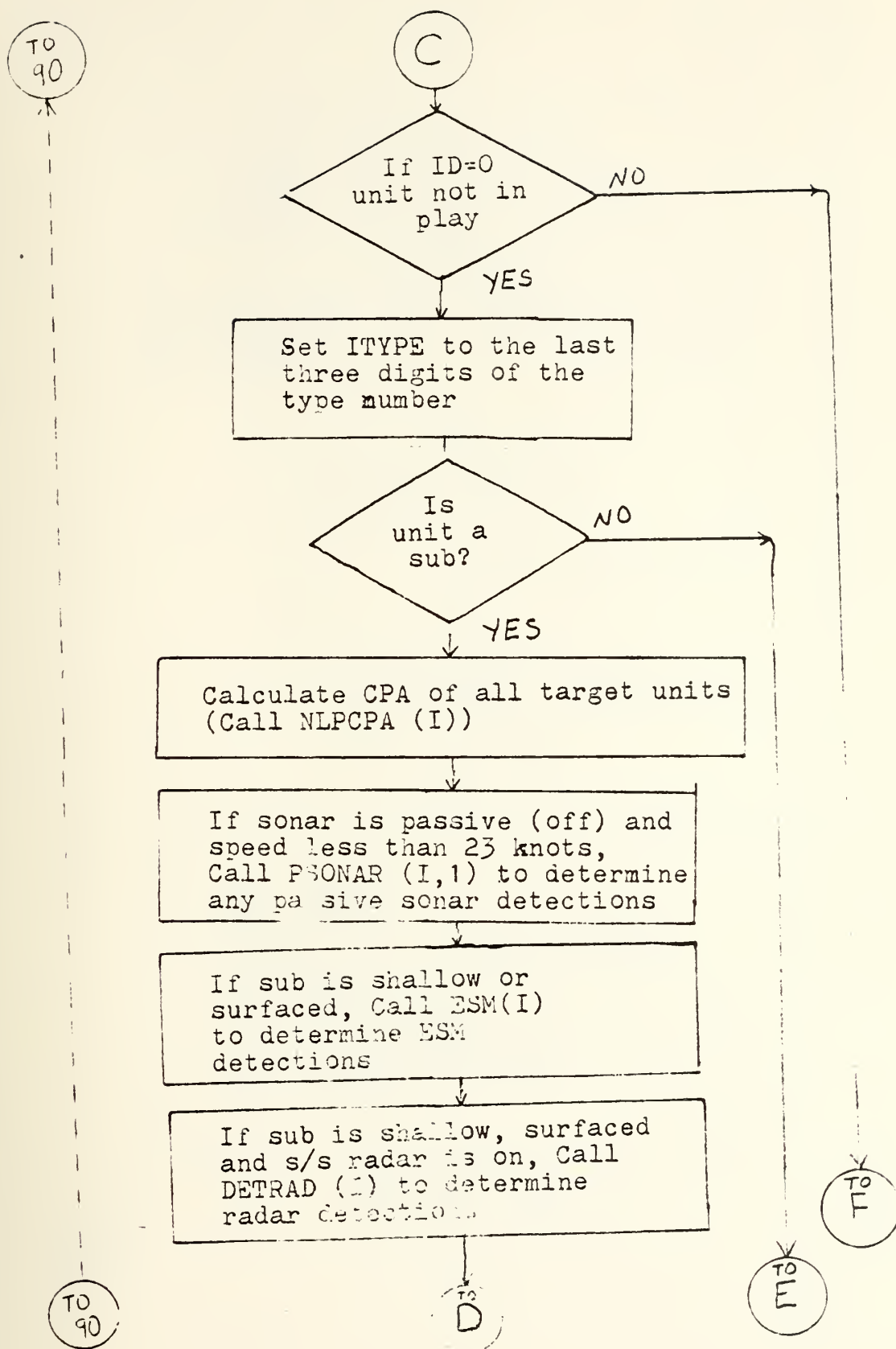
VARIABLE	DESCRIPTION
HR	Counter for six hour surveillance periods.
ITYPE	Last three digits (or last two if an aircraft) of the Ith units type number.
I	Subscript of Ith friendly unit in do loop 90, Ith enemy unit in do loop 16.
IUNIT	Total number of friendly units.
LIGHT	Set by the program to 0 = Dark, 1 = Daylight as a function of game time.
N	Counter for the number of HFDF detections.
NN	Total number of enemy units.
NX	Used in do loop 11 to calculate six hour periods for surveillance periods.
Z2	The time in minutes the previous turn ended.

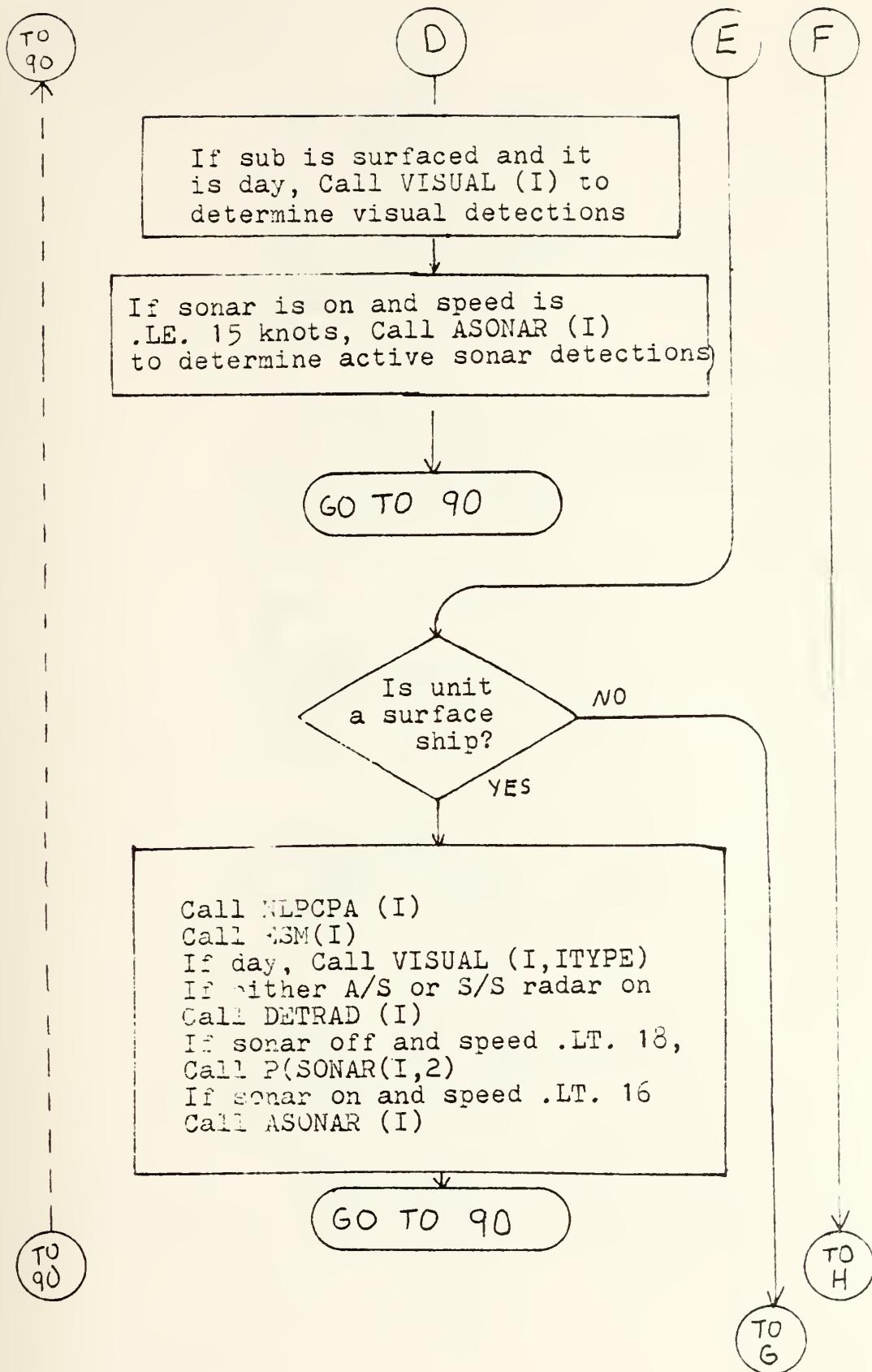


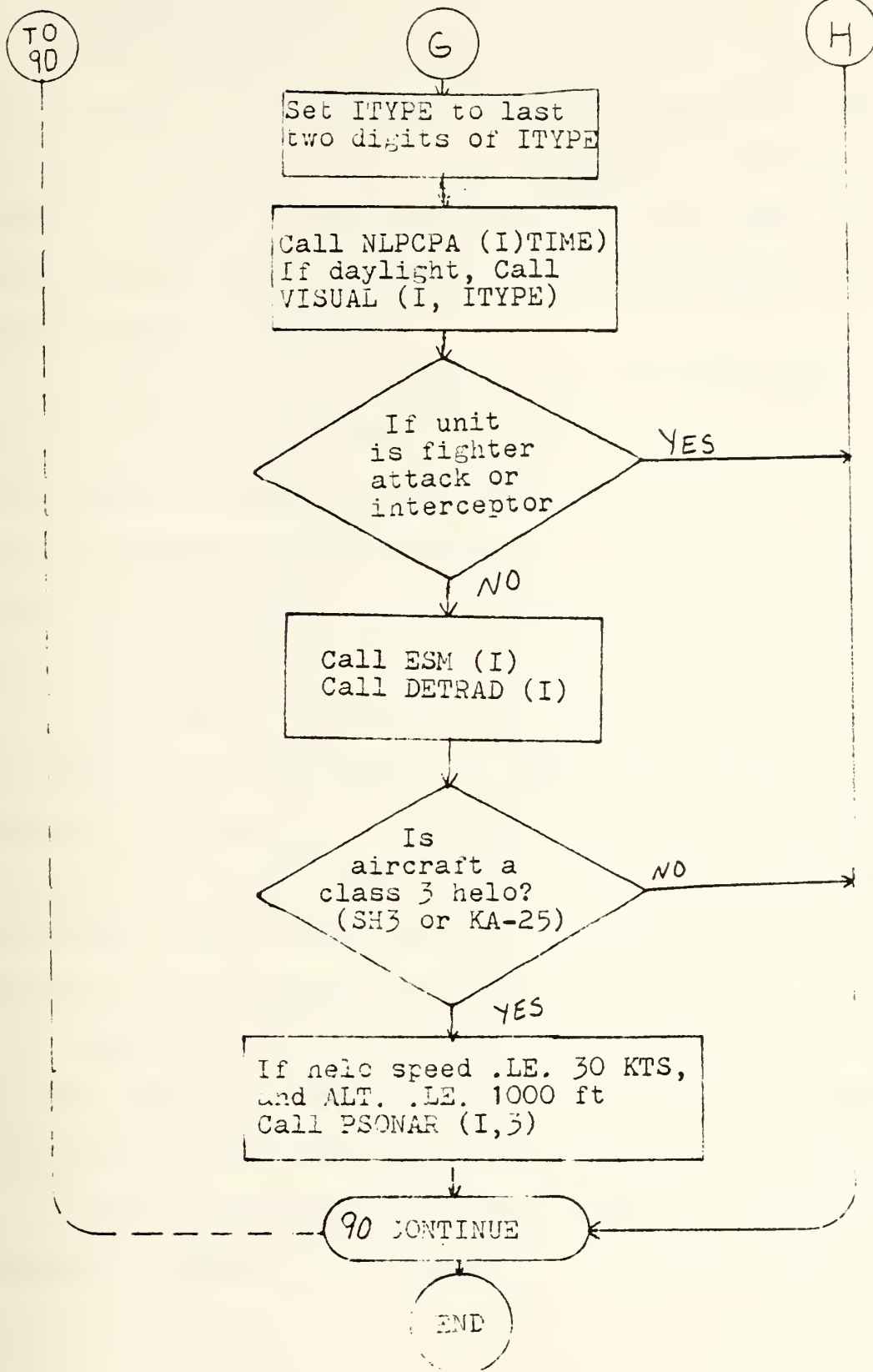
Flowchart of Subroutine DETECT

Figure 7









5. Subroutine SPOOK

This subroutine is played daily at 0001, 0601, 1201, and 1801 hours. In game terms this is an abstracted representation of sources of military intelligence available beyond the level of command represented in the game. The CASTAG program only plays level I of the three surveillance levels available in SEATAG.

All ships, submarines going over 15 knots and radiating aircraft are subject to detection. A $U(0,1)$ random number is drawn for each unit and if that unit is in one of the three preceding categories, it may be detected. If the unit is in EMCON, i.e., emitting no radiation including sonar, the chance of detection is reduced twenty percent.

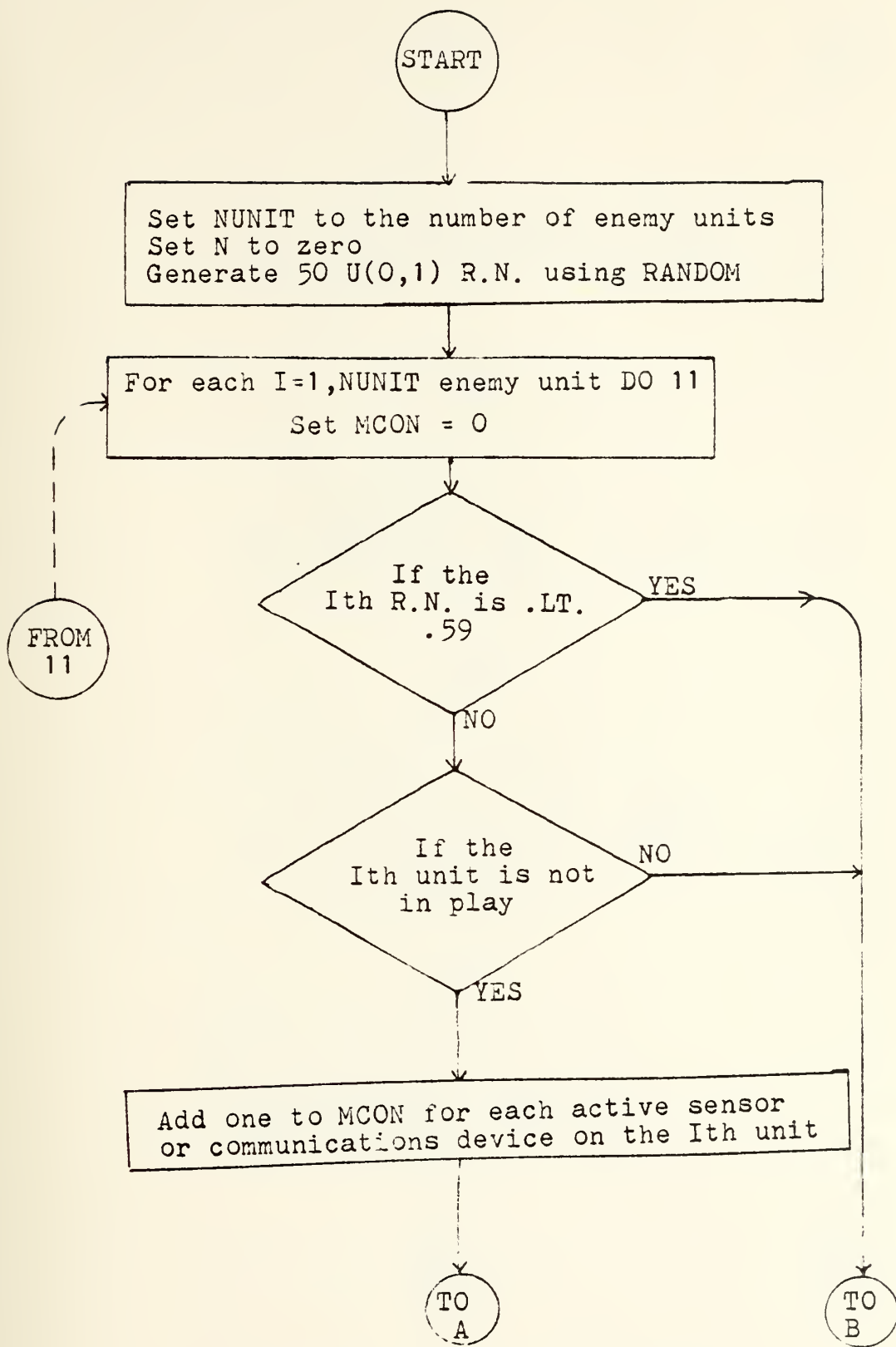
If a unit is detected, grid location and general/type information is provided. In some cases additional information consisting of the units course to the nearest 10 degrees and the last two digits of the unit's type number is provided. This information is a condensed version of Table II-5 in the SEATAG documentation.

Subroutine SPOOK is called at most once per game turn from subroutine DETECT only if the current game turn included one of the four times that surveillance is played.

Table 9 describes the local variables used in this subroutine. Figure 8 is a flowchart of subroutine SPOOK.

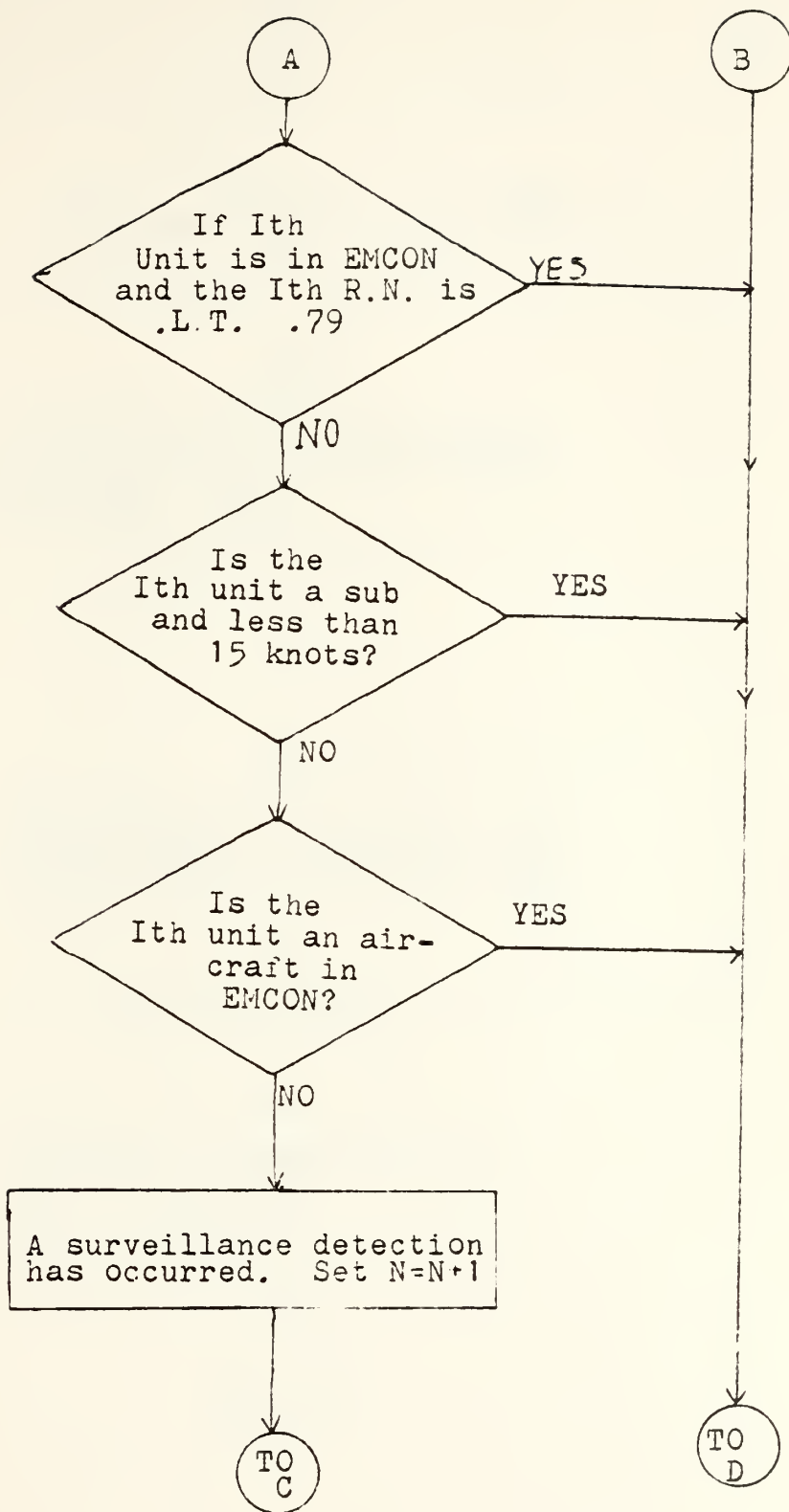
Table 9
Subroutine SPOOK, Local Variables

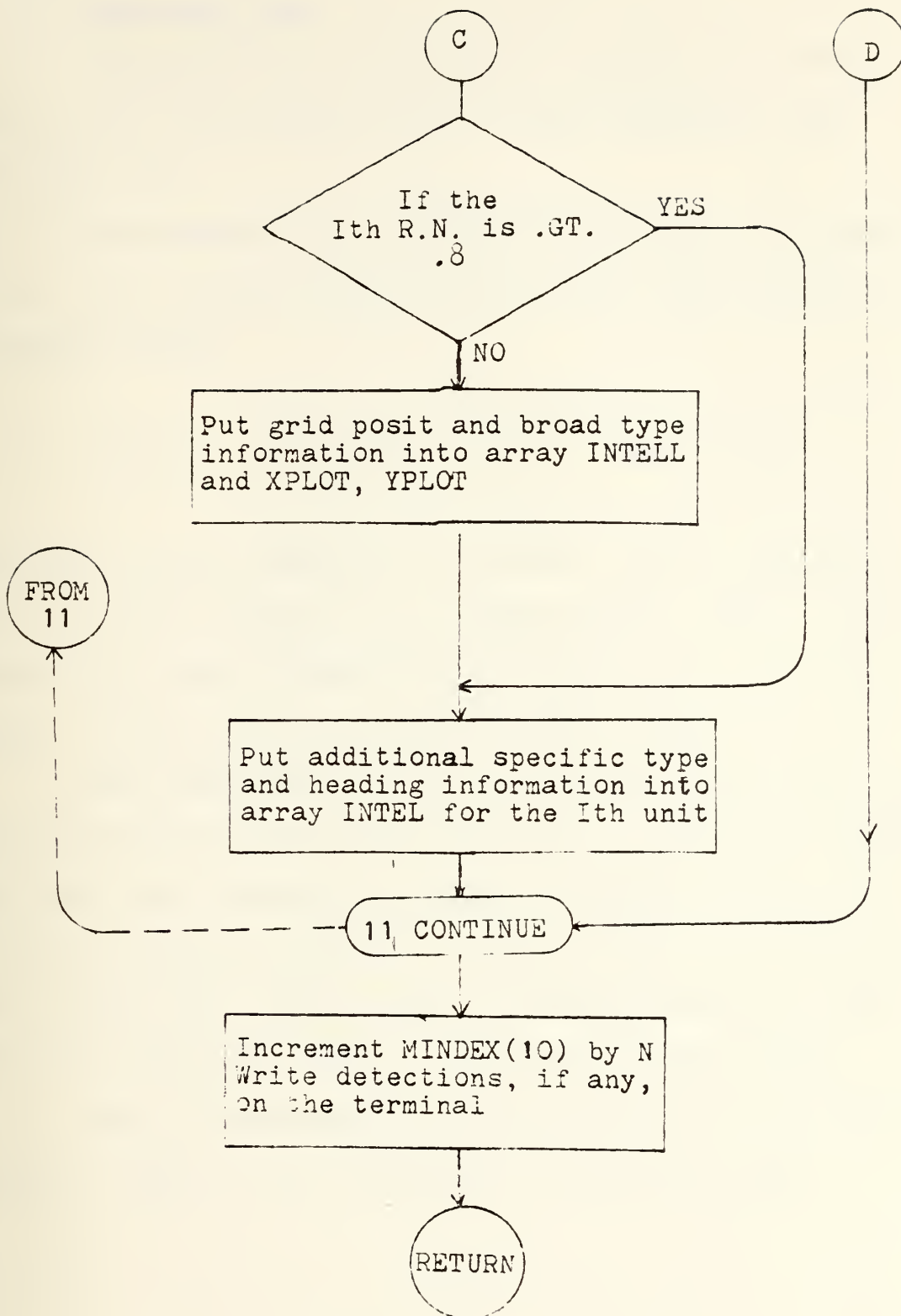
<u>Variable</u>	<u>Description</u>
I	Row subscript of the unit being checked.
ID2	Set to first two digits of the Ith unit's type number.
J	Varying column sbuscript of array DATA in do loop 9.
MCON	Sum of the number of active emitters on a unit. MCON = 0 means unit is in EMCON.
N	Counter for number of surveillance detections.
NUNIT	Set to total number of enemy units.



Flowchart of Subroutine SPOOK

Figure 3





6. Subroutine ESM

Subroutine ESM determines all possible ESM detections by the Ith friendly unit of radar emissions from each enemy unit.

The maximum range for ESM detections is calculated as twice the maximum range of the emitter times the percent of maximum range:

$$\text{Range} = 2 * \text{PRMAX} (I, N, 3) * (\text{Radar Range})$$

The subscripts for PRMAX are I denoting the Ith detecting unit, N the Nth target and 3 the ESM sensor. This is compared to the range of the CPA to determine if a detection occurs. Detections are stored in array ESMDET.

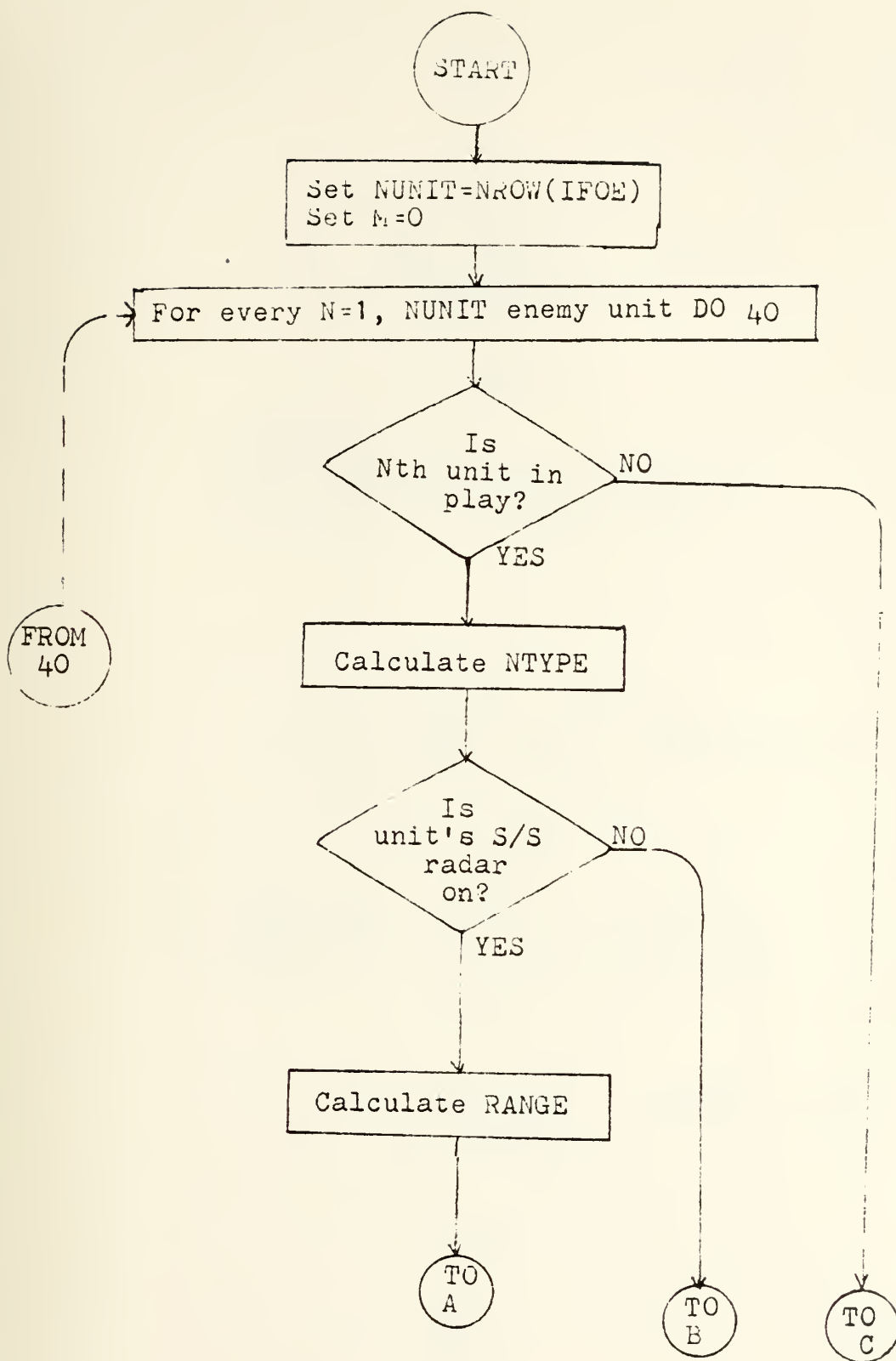
This subroutine is called from DETECT when the Ith unit is eligible to make ESM detections. Subroutine NLPCPA must have been called previously from DETECT so that the range and bearing of the CPA of every enemy unit is available in array CPA. The percent of maximum range for the Ith units ESM system must be available in array PRMAX.

Table 10 contains the local variables used in this routine. Figure 9 is a flowchart of subroutine ESM.

Table 10

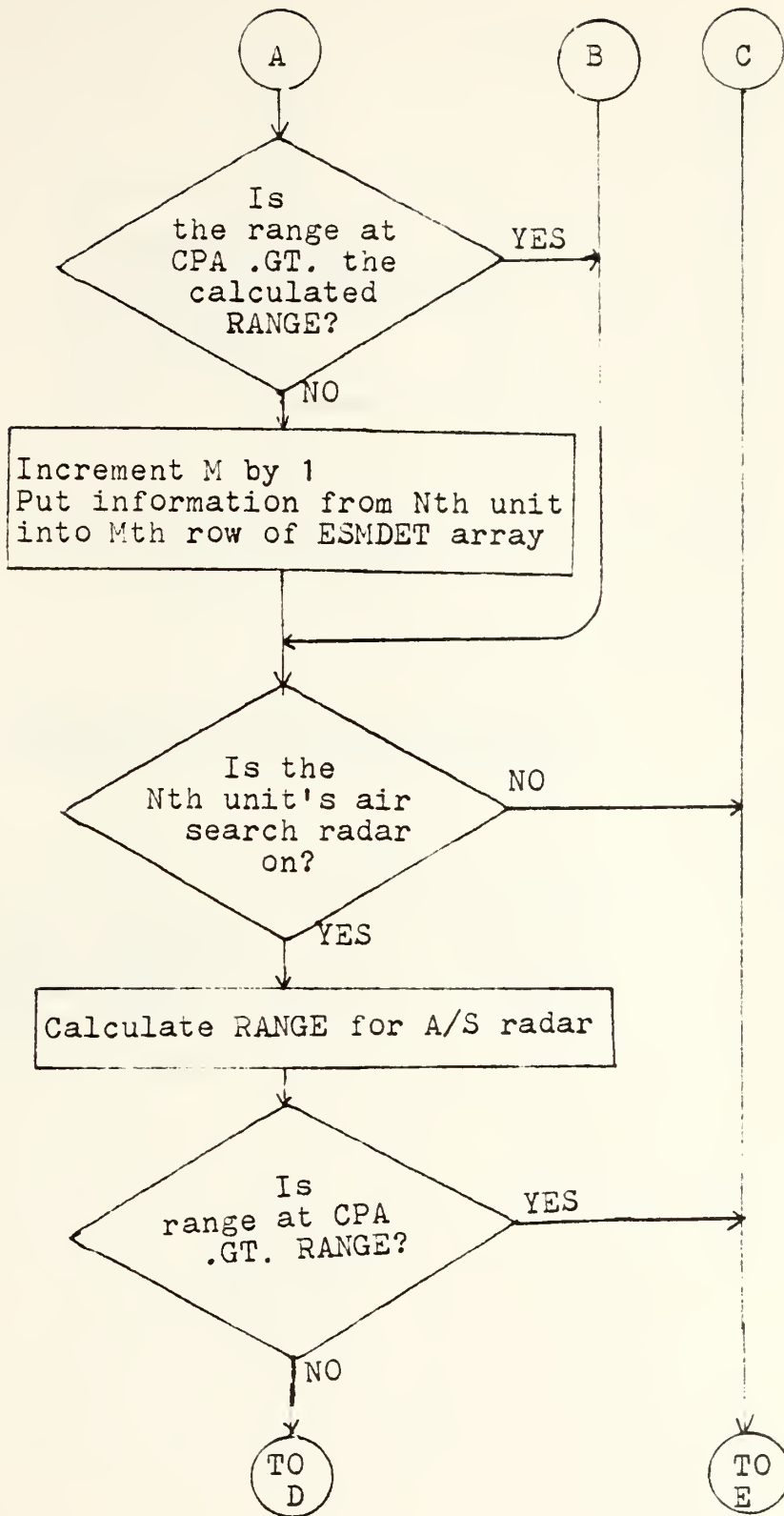
Local Variables of Subroutine ESM

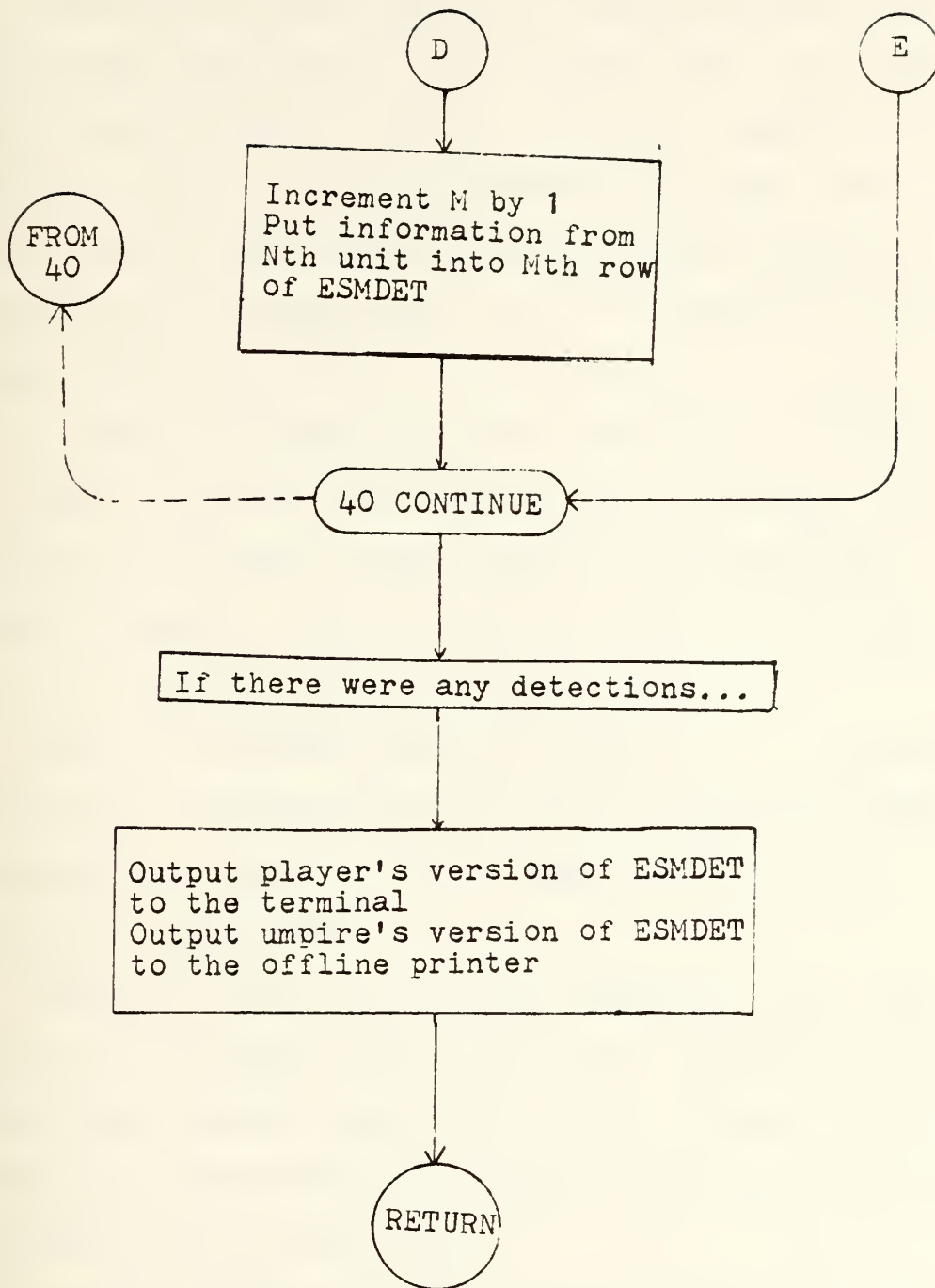
<u>Variable</u>	<u>Description</u>
I	Argument passed from DETECT denoting the Ith friendly unit as the unit attempting to make the detections.
II, JJ	Row and column subscripts for printing out array ESMDET.
M	Counter for number of ESM detections for the Ith unit.
N	Row subscript of DATA denoting the Nth enemy unit.
NTYPE	Last three digits of the Nth unit's type number.
NUNIT	Set to total number of enemy units.
RANGE	Range at which detection may take place.



Flowchart of Subroutine ESM(I)

Figure 9





7. Subroutine DETRAD

Subroutine DETRAD checks all possible radar detections of enemy units for the Ith friendly unit. Air search radars are only allowed to detect aircraft. Surface search radars only detect ships and surfaced submarines. The detection range of both air and surface search radars is calculated as the maximum range times the percent of maximum range from array PRMAX.

A detection takes place when the detection range of the radar is greater than the CPA of the target. For submarines, this range is then halved making them more difficult to detect. Detections are stored sequentially in the first ten columns of integer array RADAR.

The only argument passed to DETRAD is I. I specifies the row of the friendly unit attempting to make the radar detections and is used as the row subscript to access information in array DATA.

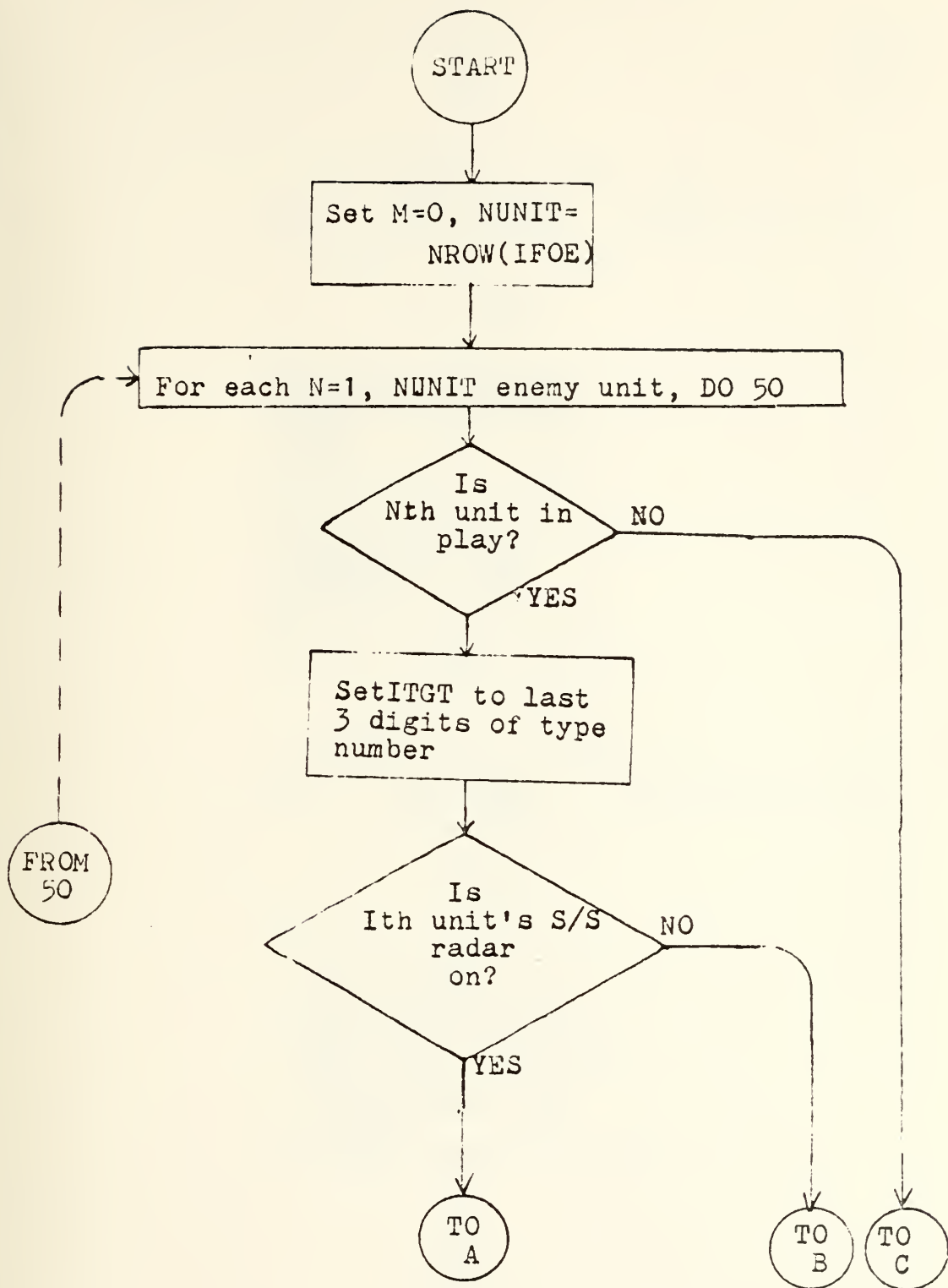
DETRAD is called from DETECT whenever a unit has either an air or surface search radar on. Subroutine NLPCPA must have been executed last for unit I. Values from PRMAX with the third subscript 1 for surface search and 2 for air search radars must be available.

Table 11 lists the local variables of DETRAD and Figure 10 is a flowchart of the routine.

Table 11

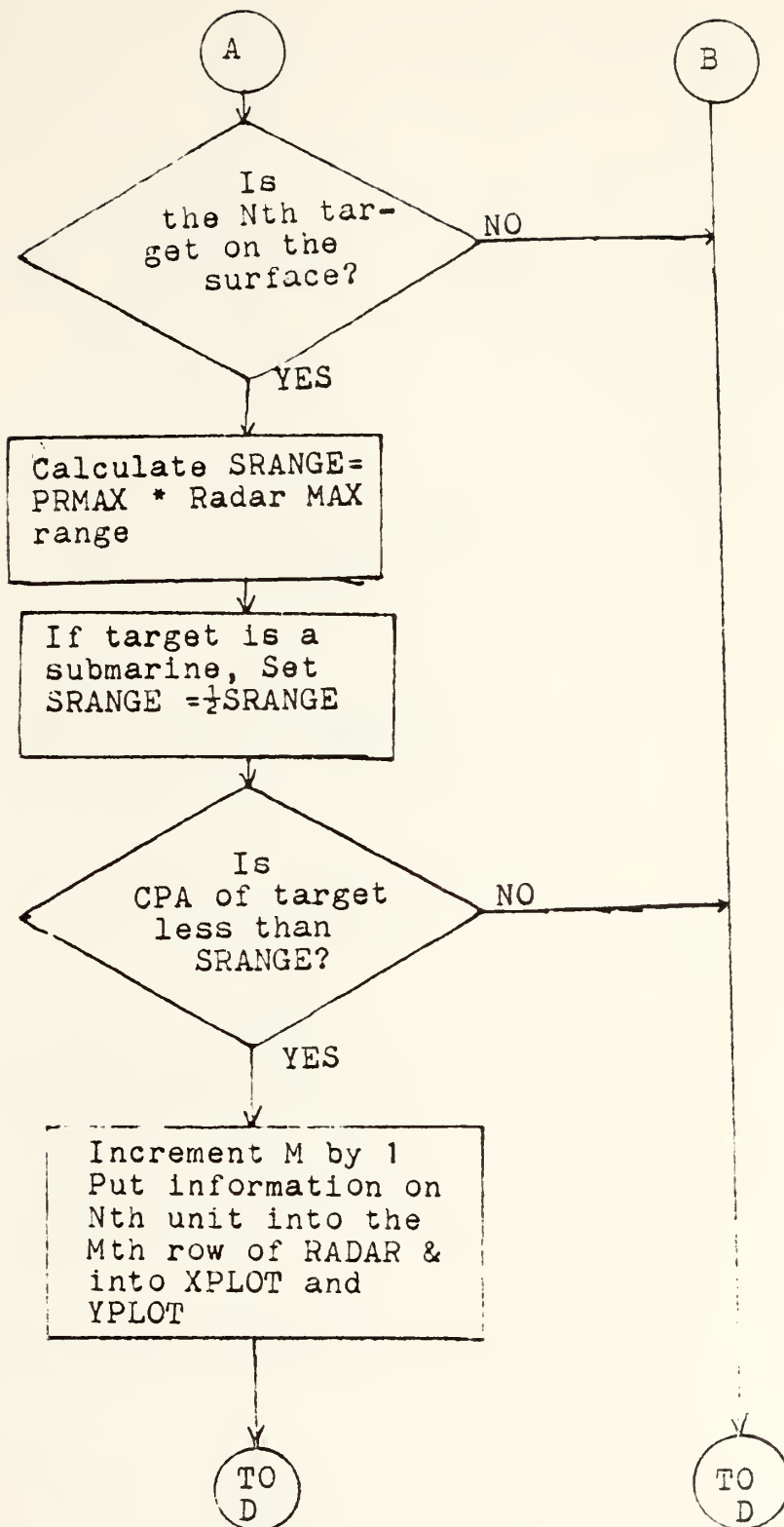
DETRAD, Local Variables

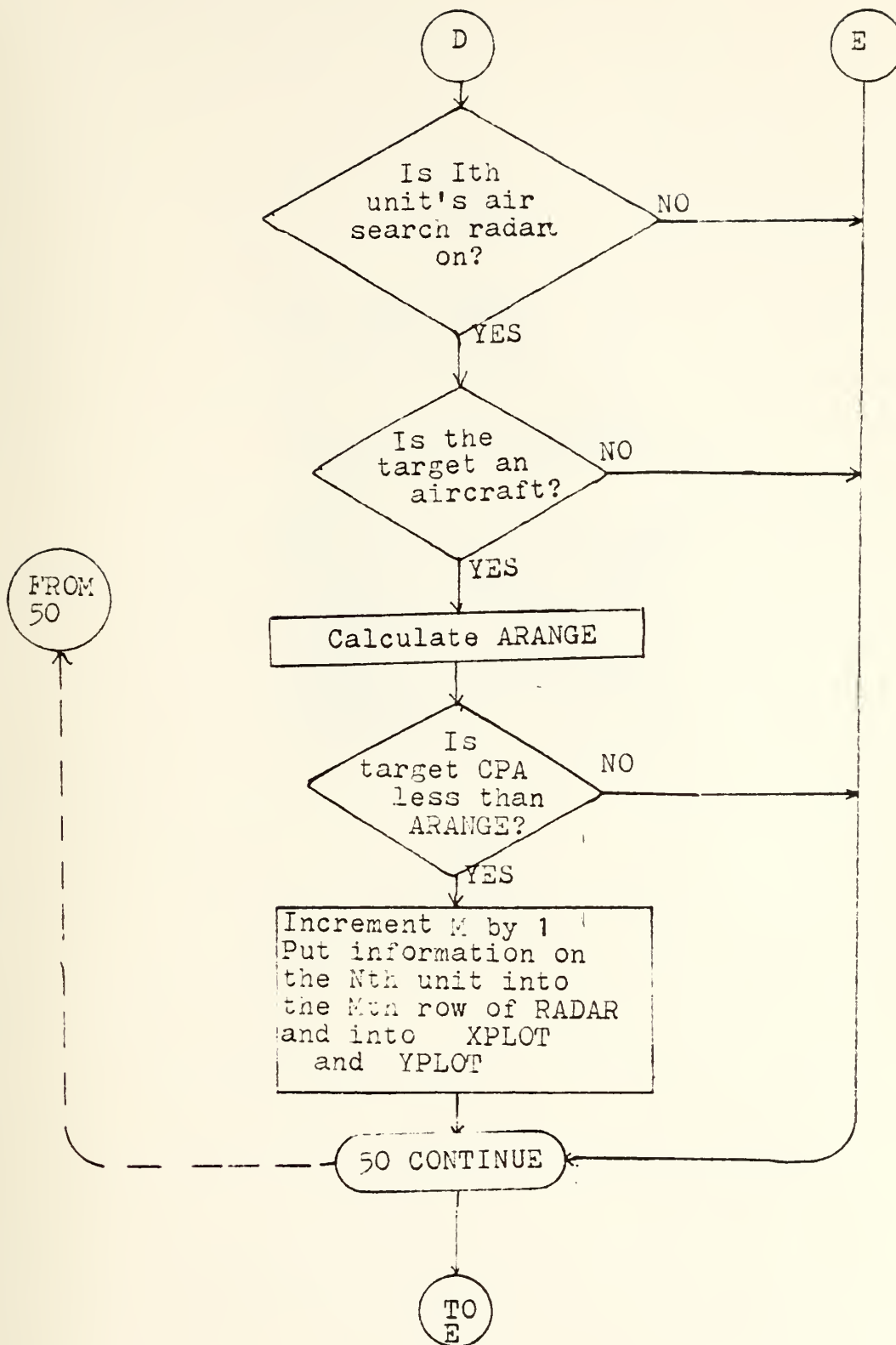
<u>Variable</u>	<u>Description</u>
ARANGE	A/S Radar calculated final detection range.
I	External argument passed from DETECT. Designates the Ith unit which is doing the detecting.
II, JJ	Row and column subscripts for printing out first 10 columns of array RADAR.
ITGT	Last three digits of the Nth unit's type number.
J	Column subscript in do loop 30 for transferring detection data to array RADAR.
M	Counter for total number of detections for unit I.
N	Subscripts the row, of the Nth enemy unit.
NUNIT	Set to total number of enemy units.
SRANGE	S/S radar calculated final detection range.

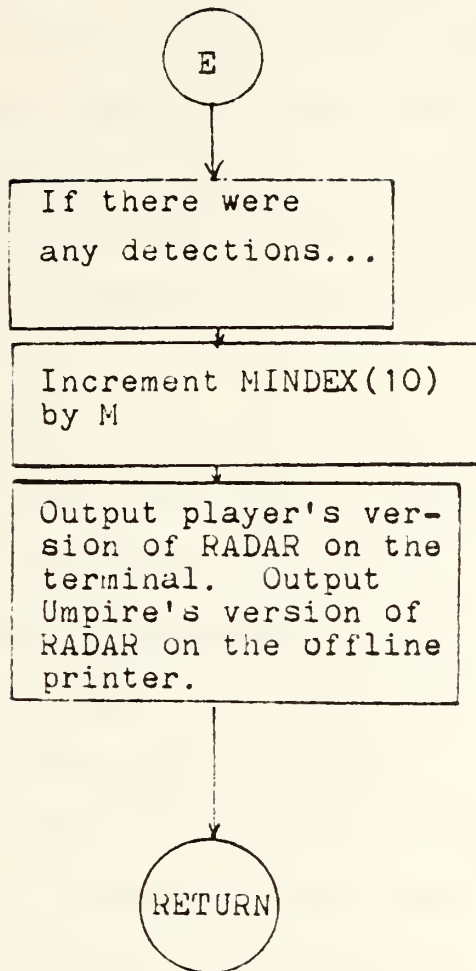


Flowchart of Subroutine DETRAD

Figure 10







8. Subroutine VISUAL

Subroutine VISUAL processes all possible visual detections of enemy units for the Ith friendly unit. Ships and surfaced submarines were arbitrarily assigned an unmodified visual detection range of 10 n.m. Aircraft were assigned a 20 n.m. range. Although an aircraft or ship can discern a target at more than the above ranges, visual detection within the framework of SEATAG implies classification opportunities as well.

Classification implies the aircraft or ship closes to read hull numbers and count masts. Rather than try to make the information provided a function of range, 10 and 20 miles were selected as reasonable values considering the aggregate level of the game. All units with the exception of submerged subs are detectable by this routine.

A detection occurs when the calculated CPA of the target is less than the assigned visual detection range. The information on the detection is stored in columns 11-20 of array RADAR.

The two arguments passed to VISUAL are I which designates the row in array DATA of the unit attempting to make the detection and ITYPE which tells the subroutine if the Ith unit is an aircraft or surface unit.

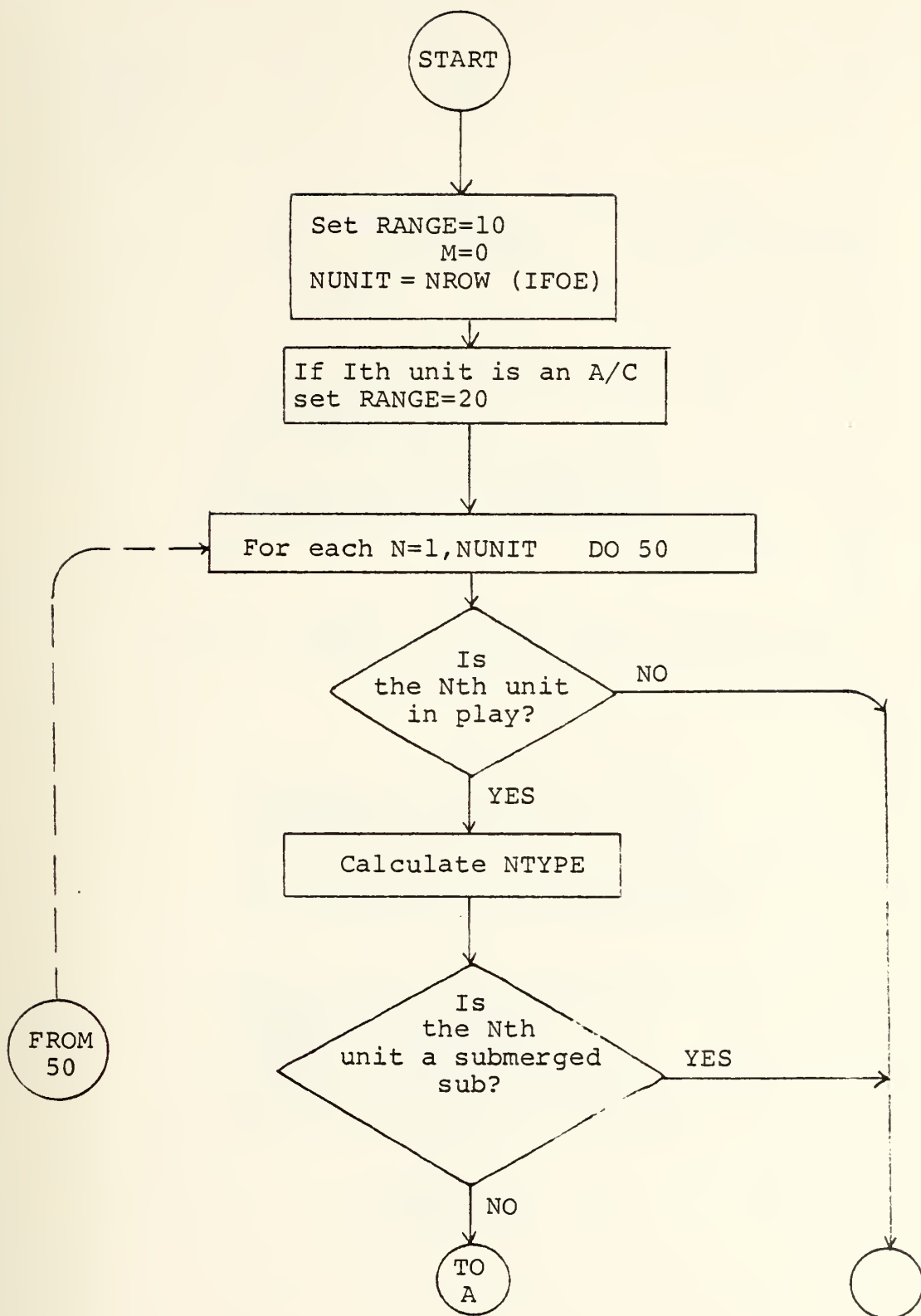
This subroutine is called from DETECT when the Ith unit is an aircraft, ship, or a surfaced submarine and game time is between 0600 and 1800 hours. Subroutine NLPCPA must have been executed last for unit I.

Table 12 lists the local variables and Figure 11 is a flowchart of subroutine VISUAL.

Table 12

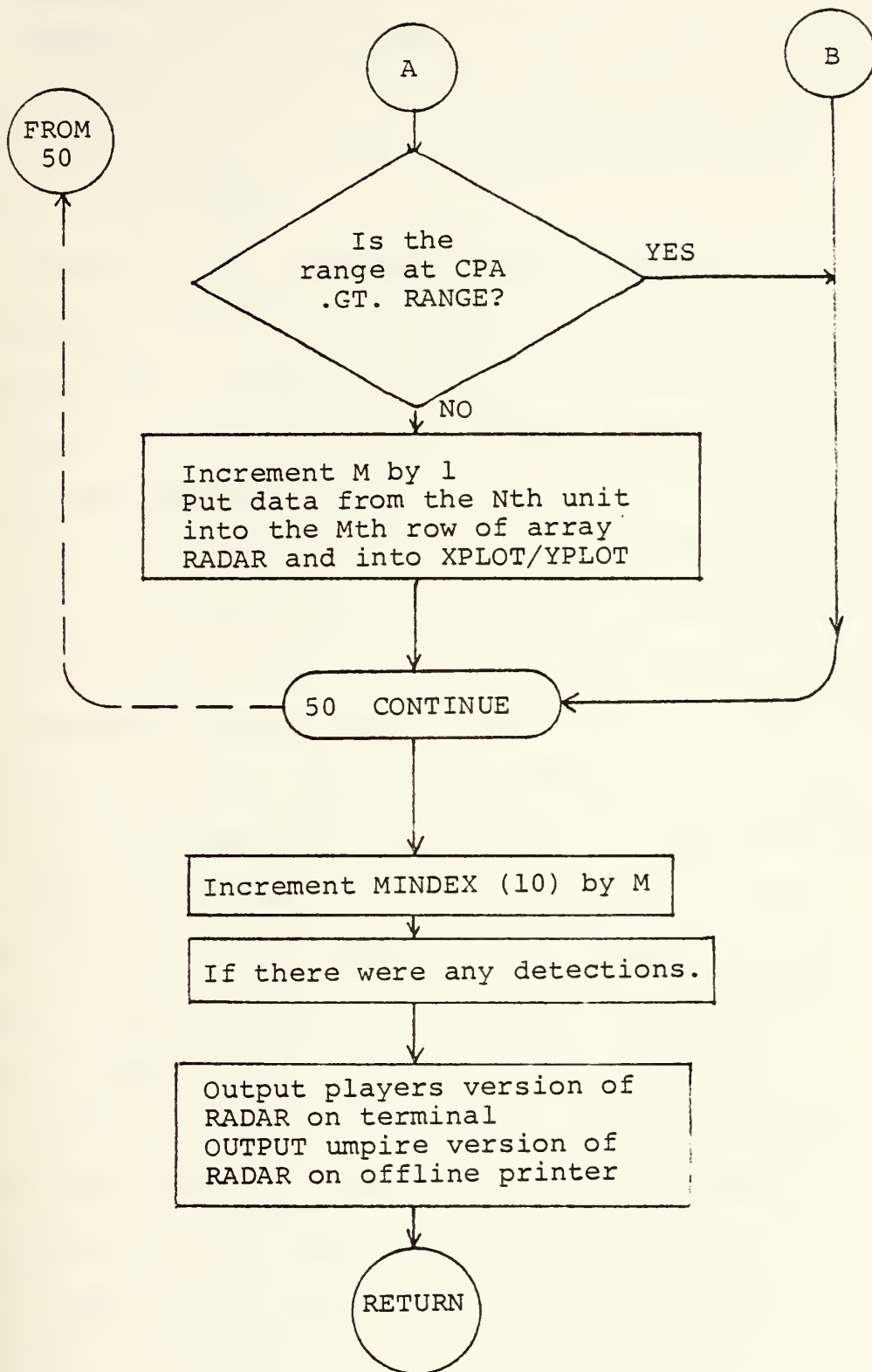
Subroutine VISUAL, Local Variables

<u>Variable</u>	<u>Description</u>
I	Argument passed from DETECT denoting the Ith unit doing the searching.
II, JJ	Row and column subscripts for printing out columns 11-20 of array RADAR.
ITYPE	Last two digits of Ith unit type number if aircraft, last three digits if unit submarine or ship.
M	Counter for number of visual detections for the Ith unit.
N	Row subscript denoting the Nth enemy unit.
NTYPE	Last three digits of Nth unit's type number.
NUNIT	Total number of enemy units.
RANGE	Set to 10 n.m. if Ith unit on surface, 20 n.m. if Ith unit an aircraft.



Flowchart of Subroutine VISUAL

Figure 11



9. Subroutine ASONAR

ASONAR, for active sonar, determines possible active sonar detections by the Ith friendly unit. The unit's sonar must be active and the unit's speed less than 15 knots prior to calling ASONAR.

Surface ships, submarines and helicopters with dipping sonars may make active detections. Only submerged submarines, either friendly or enemy, may be detected by an active sonar. No differentiation is made between a friendly or enemy submarine detection.

Level 5 of PRMAX is used for active sonar detections of own units although it is normally used in determining passive performance. This is to avoid correlating active sonar performance against the Nth enemy unit with that against the Nth friendly unit. This avoids adding another level ($K = 6$) to array PRMAX for what amounts to a rare occurrence.

The only argument passed to ASONAR is I. This specifies the row of the friendly unit attempting to make the active sonar detections and is used as the row subscript to access the Ith unit's data in array DATA.

ASONAR is called from DETECT. Values for percent of maximum range from array PRMAX with the third subscript equal to four and five are required before the subroutine is called. NLPCPA must have been called last for the Ith unit prior to calling ASONAR.

Prior to the second time through do loop 60, NLPCPA is called from ASONAR. After IFOE is set to ID, NLPCPA

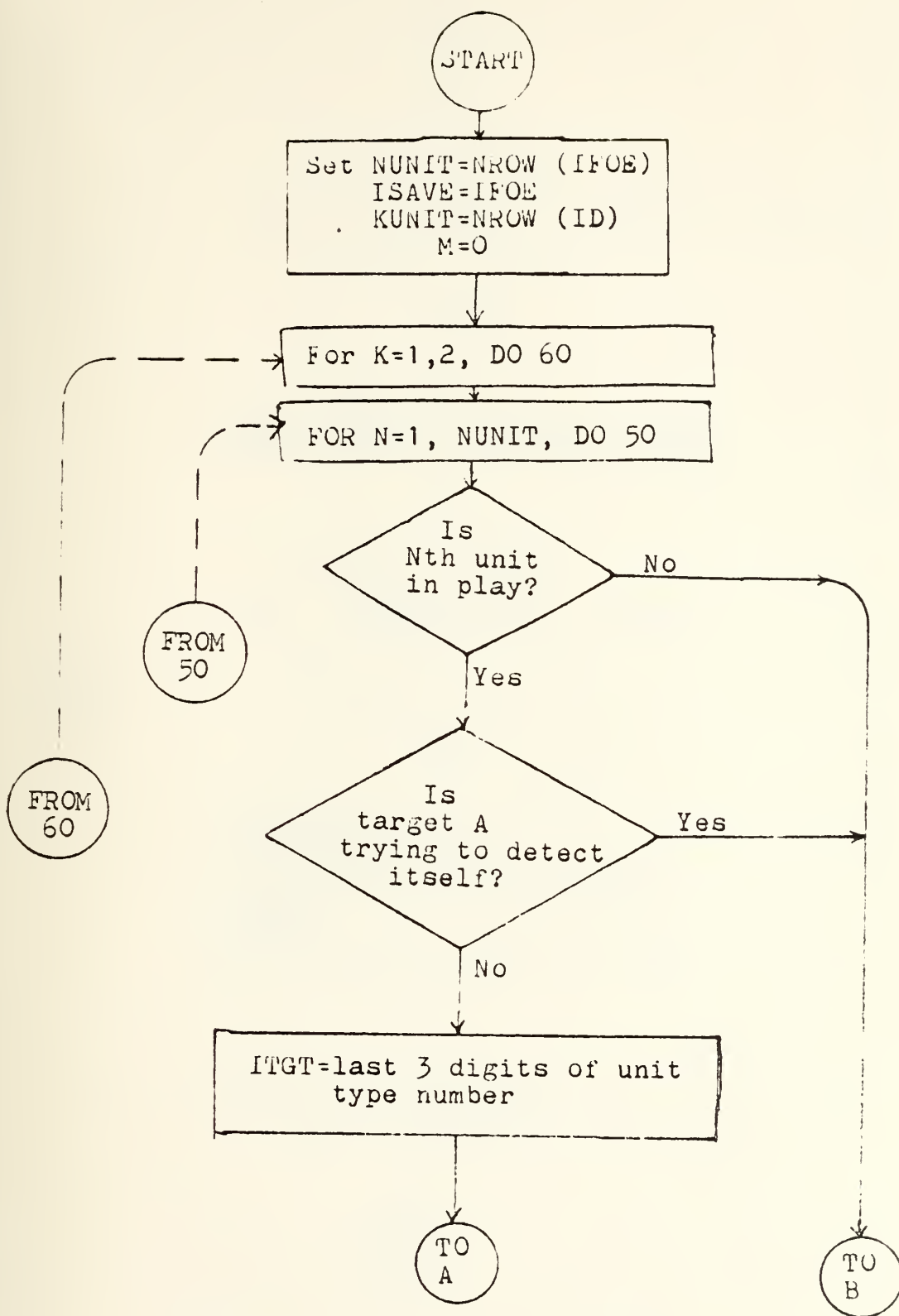
calculates the CPA of all friendly units and stores the values in array CPA. This is the only instance of a subroutine being called from a subroutine other than from DETECT in program SEARCH. Note that ASONAR must be the last subroutine called for detections by the Ith unit in DETECT due to its calling NLPCPA as the values in array CPA will be those of friendly units.

Detections for the Ith unit are stored in the first ten columns of array SONAR by row. Table 13 lists the local variables of ASONAR. Figure 12 is a flowchart of ASONAR.

Table 13

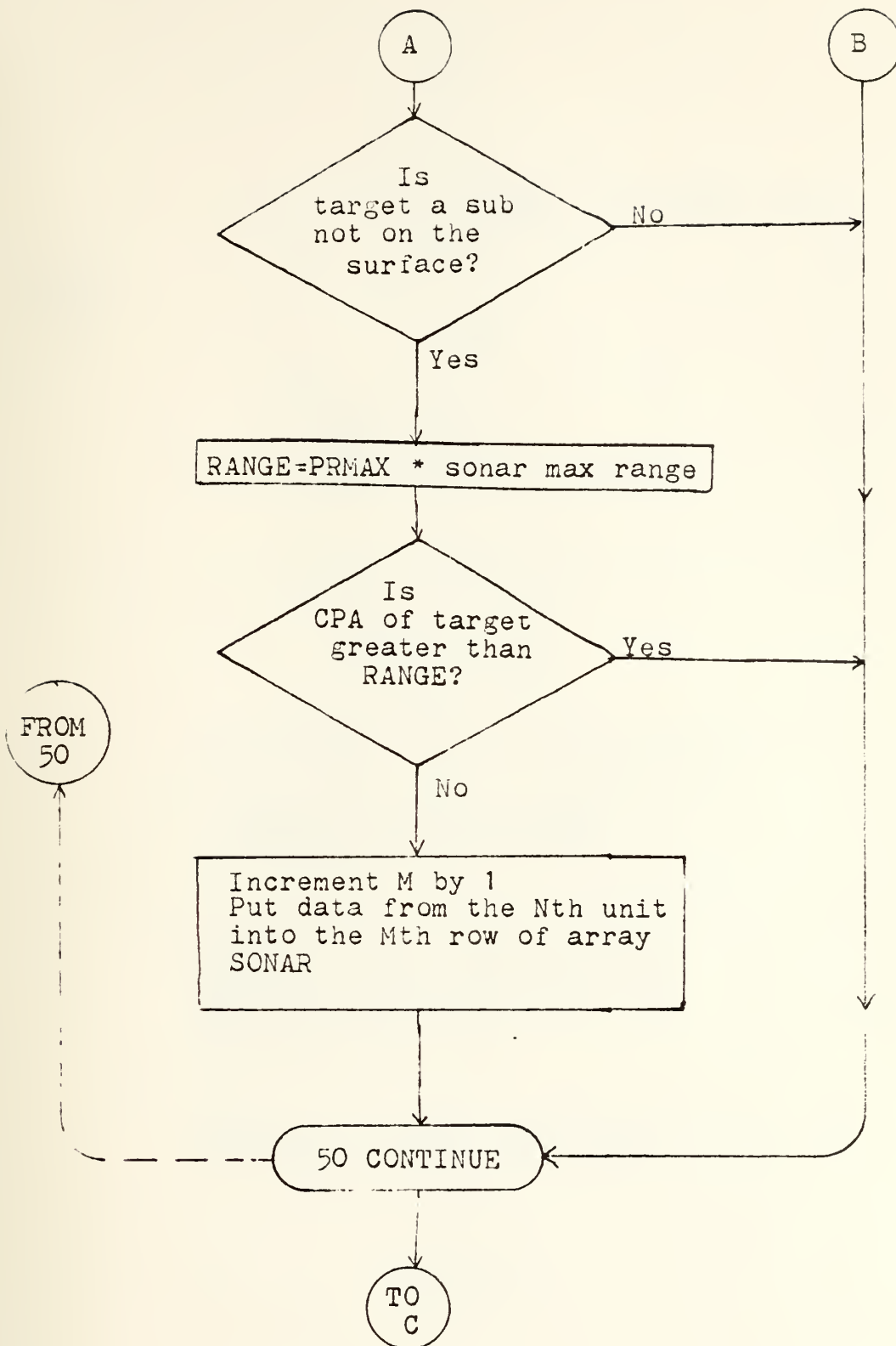
Subroutine ASONAR, Local Variables

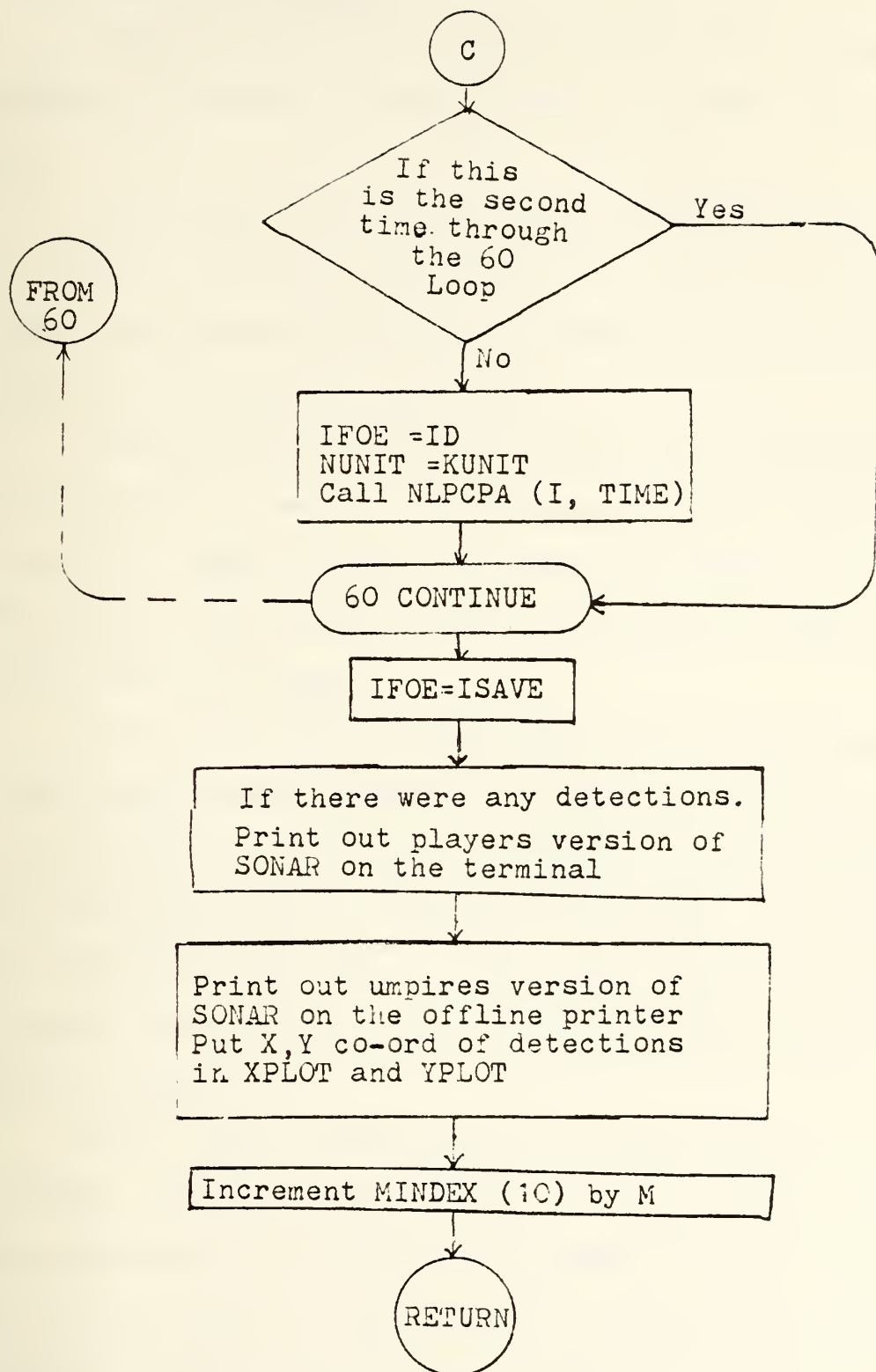
<u>Variable</u>	<u>Description</u>
I	Argument passed from DETECT denoting the Ith unit attempting active sonar detections.
IA	Column subscript in do loop 40 to transfer detection data to array SONAR.
II, JJ	Row and column subscripts for printing out array SONAR.
ISAVE	Saves value of IFOE while IFOE = ID.
ITGT	Last three digits of the Nth unit's type number.
K	Counts twice through do loop 60.
KUNIT	Set to total number of friendly units.
M	Counter for total number of active sonar for the Ith unit.
N	Row subscript denoting the Nth enemy when K = 1 or friendly when K = 2 unit.
NUNIT	Set to total number of enemy units when K = 1, total number of friendly units when K = 2.
RANGE	Active sonar calculated final maximum detection range.
TIME	Argument passed to NLPCPA denoting this turn's length in minutes.



Flowchart of Subroutine ASONAK

Figure 12





10. Subroutine NLPCPA

NLPCPA calculates the range in n.m. and the bearing in degrees of the CPA of every enemy unit to the Ith friendly unit.

The final CPA is in fact the CPA during the last turn. In most instances this turns out to be the initial or final position of the Nth unit, depending on if that unit is opening or closing the Ith unit.

The subroutine uses each unit's (X,Y) coordinates, course and speed as well as the elapsed time and clock time of this turn. Recall that since subroutine MOVE is called before NLPCPA, the values of the coordinates are where the units are after the move.

The method used in this subroutine to determine the CPA takes the approach commonly used underway where the relative position of the target is known and the target's course, speed and CPA are determined from that. This will appear somewhat round about as the target course and speed are already available and a normal trigonometric approach would be feasible.

Point (A,B) is the position of the Ith unit at time ZTIME. (X2,Y2) is the relative as well as the geographic position of the Nth unit at ZTIME. Both of these points are available in array DATA. A previous relative position of the Nth unit to the Ith unit must be calculated. This point (X1,Y1), will be calculated as the relative position of the target at time (ZTIME-TIME), or the game time

at the beginning of this turn. $(X1,Y1)$ is not the position of the target at the beginning of this turn. It is a relative position at which the target may never have been. A geometric diagram of the calculation of $(X1,Y1)$ is provided as Figure 13.

To calculate $(X1,Y1)$ the X and Y distance traveled by the target is subtracted from point $(X2,Y2)$ giving the position of the target at the start of this turn. Then the X and Y distance traveled by the Ith unit is added, resulting in $X1$ and $Y1$:

$$X1 = X2 - D1 * (\sin(\text{THETA})) + \text{XDIST} \quad (1)$$

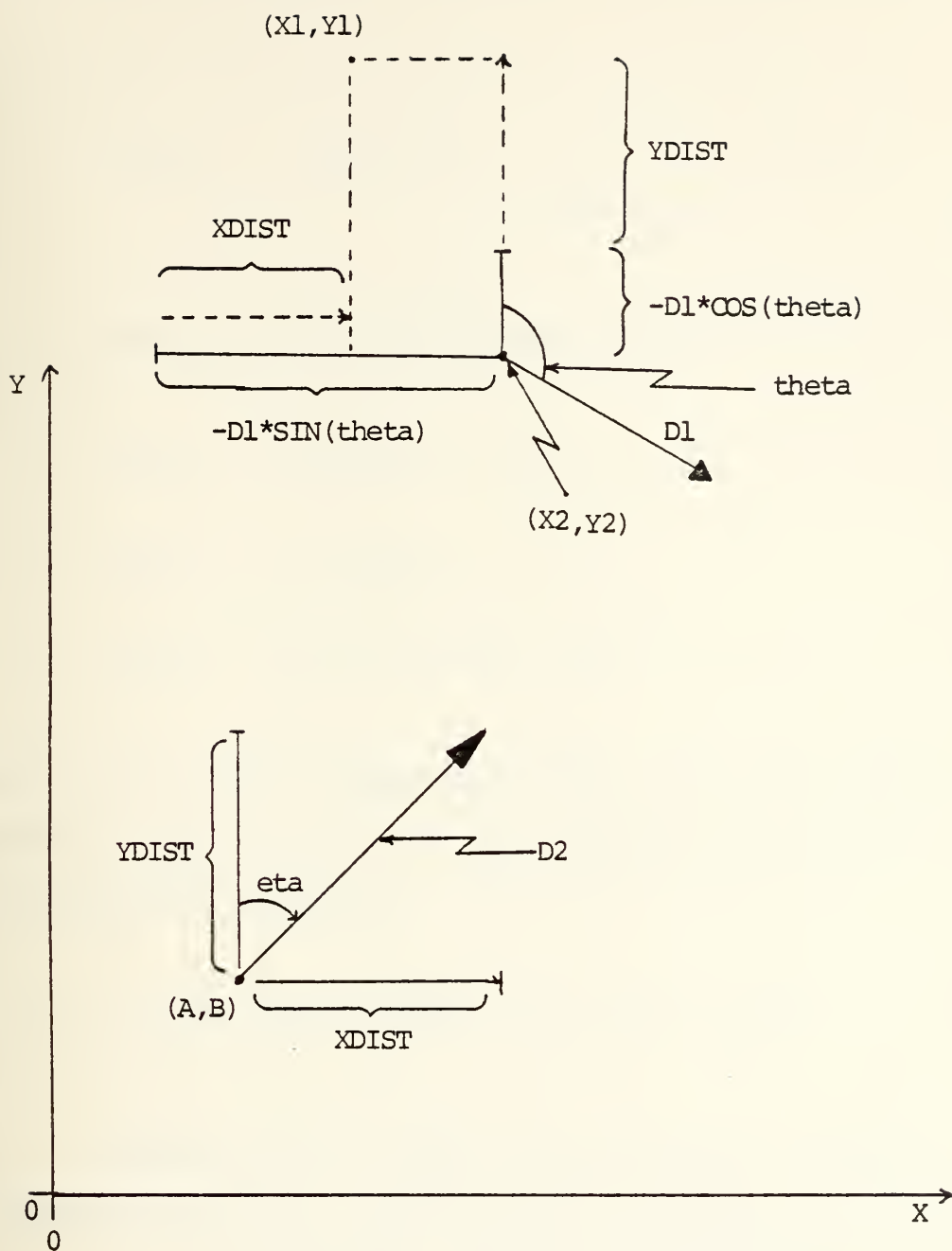
$$Y1 = Y2 - D1 * (\sin(\text{THETA})) + \text{YDIST} \quad (2)$$

To determine the CPA, the problem is to find the shortest distance from point (A,B) to the line defined by the two relative points $(X1,Y1)$ and $(X2,Y2)$. This line is called the DRM for Direction of Relative Movement. The slope is called DX . Formulating the problem as a non-linear programming problem gives

$$\text{MIN } f(x,y) = \text{SQRT}((X - A)^2 + (Y - B)^2) \quad (3)$$

$$\text{subject to} \quad Y - Y1 = DX * (X - X1) \quad (4)$$

Using the LaGrange multiplier technique, we minimize the square function minus lambda times the constraint



$$X_1 = X_2 - XDIST - D_1 \sin(\theta)$$

$$Y_1 = Y_2 - YDIST - D_1 \cos(\theta)$$

CALCULATION OF (X_1, Y_1)

Figure 13

(Eq. 5) which will have the same minimum as Equation 3.

$$F(x,y) = (X - A)^2 + (Y - B)^2 - L * (Y - Y1 - DX(X - X1)) \quad (5)$$

Taking partial derivatives yields

$$\partial F / \partial X = 2X - 2A + L * DX = 0 \quad (6)$$

$$\partial F / \partial Y = 2Y - 2B - L = 0 \quad (7)$$

$$\partial F / \partial L = DX * (X - X1) + Y1 - Y = 0 \quad (8)$$

Equation (7) gives an expression for L and (8) gives an expression for Y in terms of X. Substituting these in (6) and solving for X yields

$$X = (A + B * DX + X1 * DX^2 - Y1 * DX) / (1 + DX^2) \quad (9)$$

Equation (9) is how X is calculated in NLPCPA. Y is computed from (8). (X,Y) is the coordinate of the CPA.

In the calculation of DX, if the absolute value of (X2 - X1) is within .001 of zero, the CPA calculation is not needed since DX is infinite or nearly so. Practically speaking in this case, X2 = X1 and the DRM is vertical, yielding a CPA of (X2,B).

If both $(X2 - X1)$ and $(Y2 - Y1)$ are less than EPS, then $X2 = X1$ and $Y2 = Y1$ implying the two units involved are on the same course and speed. The CPA is set to $(X2, Y2)$ in this case.

In Figure 13, the CPA falls conveniently between $(X1, Y1)$ and $(X2, Y2)$. In fact we have three cases to consider in deciding on the final CPA. Figure 14 shows the three possibilities. The relative positions of $(X1, Y1)$ and $(X2, Y2)$ may or may not be reversed, giving six possibilities to check.

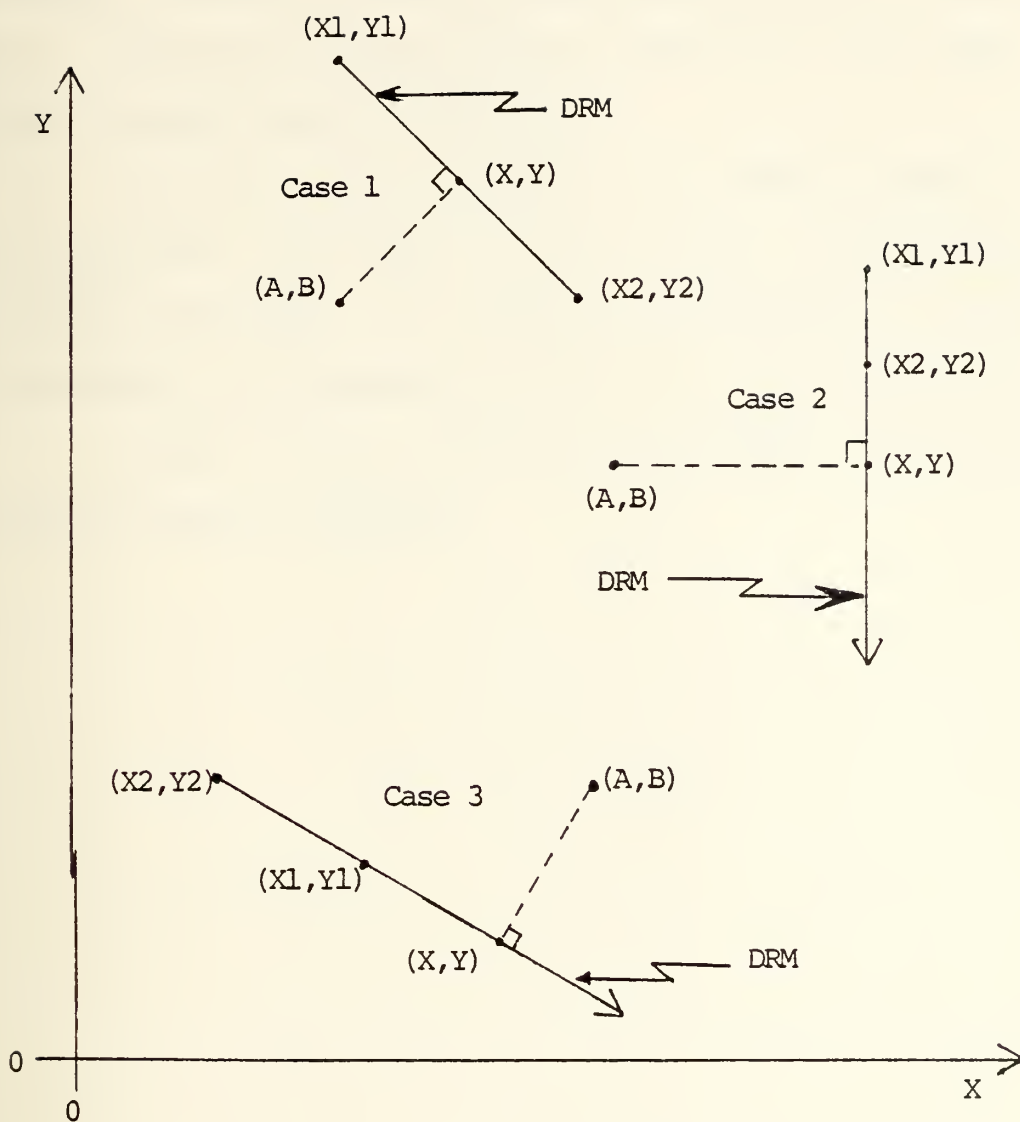
Two series of four logical FORTRAN IF statements branch to assign the appropriate final value for (X, Y) .

Once the final (X, Y) is determined, the distance from (X, Y) to (A, B) is calculated using the standard cartesian distance formula. Distances less than EPS are set to EPS to avoid dividing by zero when calculating the bearing of the CPA. The CPA range for the Nth unit is stored in $CPA(N, 1)$.

Finally the bearing of the point (X, Y) from (A, B) must be calculated and converted to degrees. TAU is the angle between line segment $((A, B), (X, Y))$ and $((X, B), (X, Y))$ and is calculated using the ARSIN function as

$$TAU = \text{ARCSIN}((X - A)/CPA(N, 1)) \quad (10)$$

In the first quadrant TAU will be the correct bearing. If (X, Y) is in one of the other three quadrants



Case	CPA Point
1	(X,Y)
2	(X_2,Y_2)
3	(X_1,Y_1)

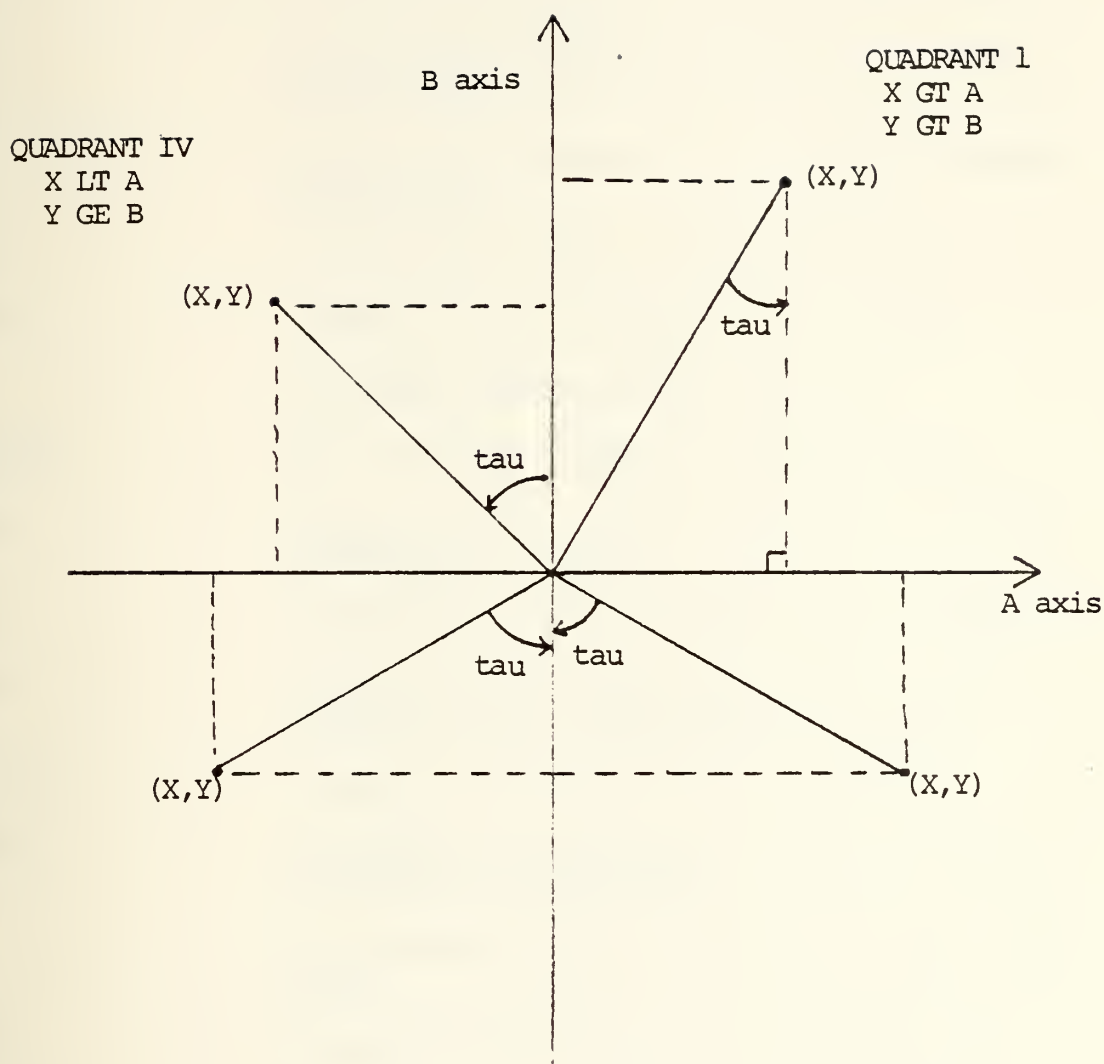
Relative Locations of DRM Endpoints

Figure 14

of (A,B) TAU must be modified as shown in Figure 15. Finally TAU is converted to degrees and stored in CPA(N,2). The argument I passed to NLPCPA specifies the row of the friendly unit. The CPA's are calculated relative to this unit. TIME is the elapsed time for this turn.

This subroutine is called from DETECT once for each Ith (friendly) unit that is able to make any ESM, radar or sonar detections. In addition, NLPCPA may be called from ASONAR in circumstances defined in that subroutine.

Figure 16 is a flowchart for NLPCPA and Table 14 lists the local variables.



Four Quadrant Tau Calculation

Figure 15

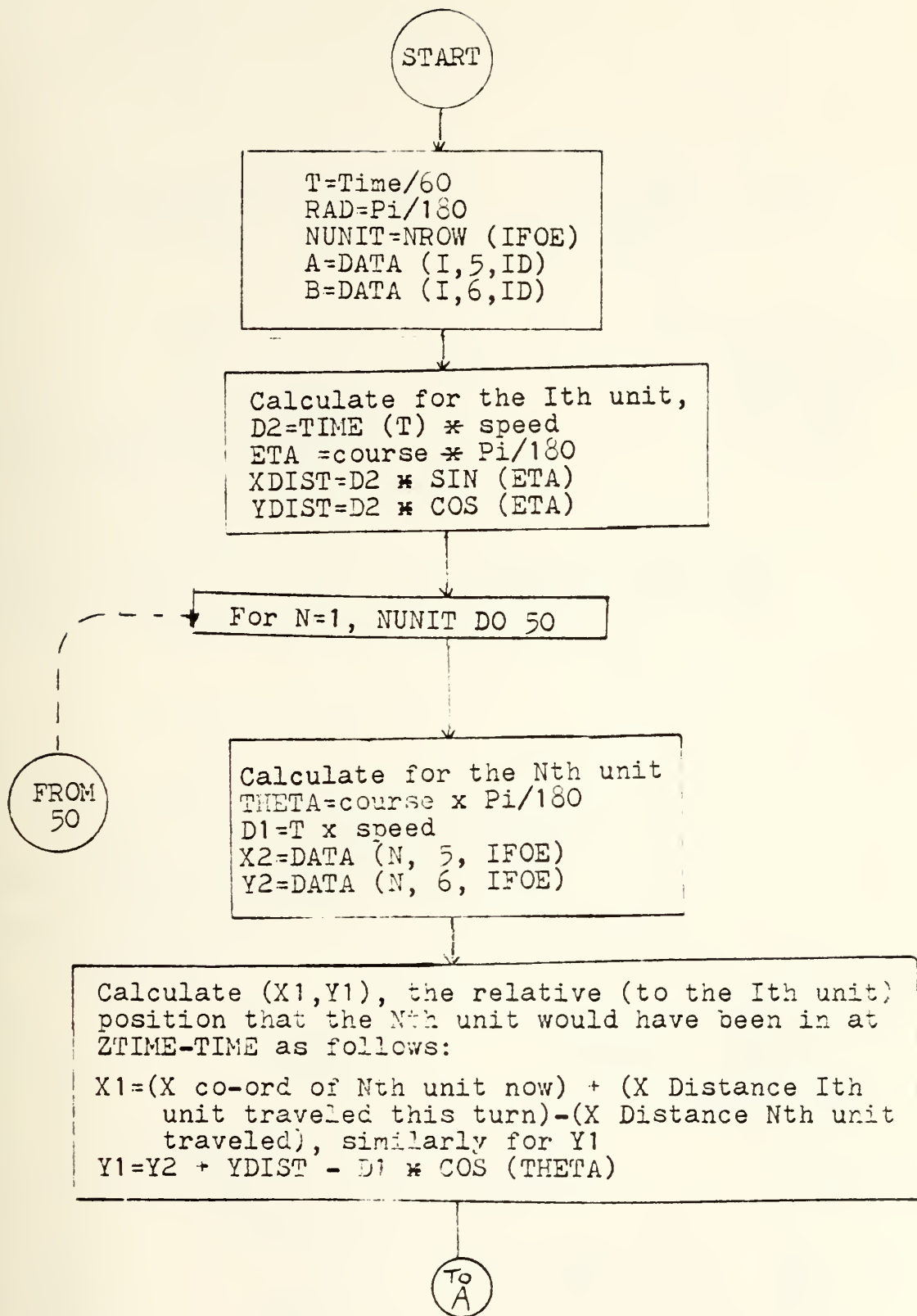
Table 14

Subroutine NLPCPA, Local Variables

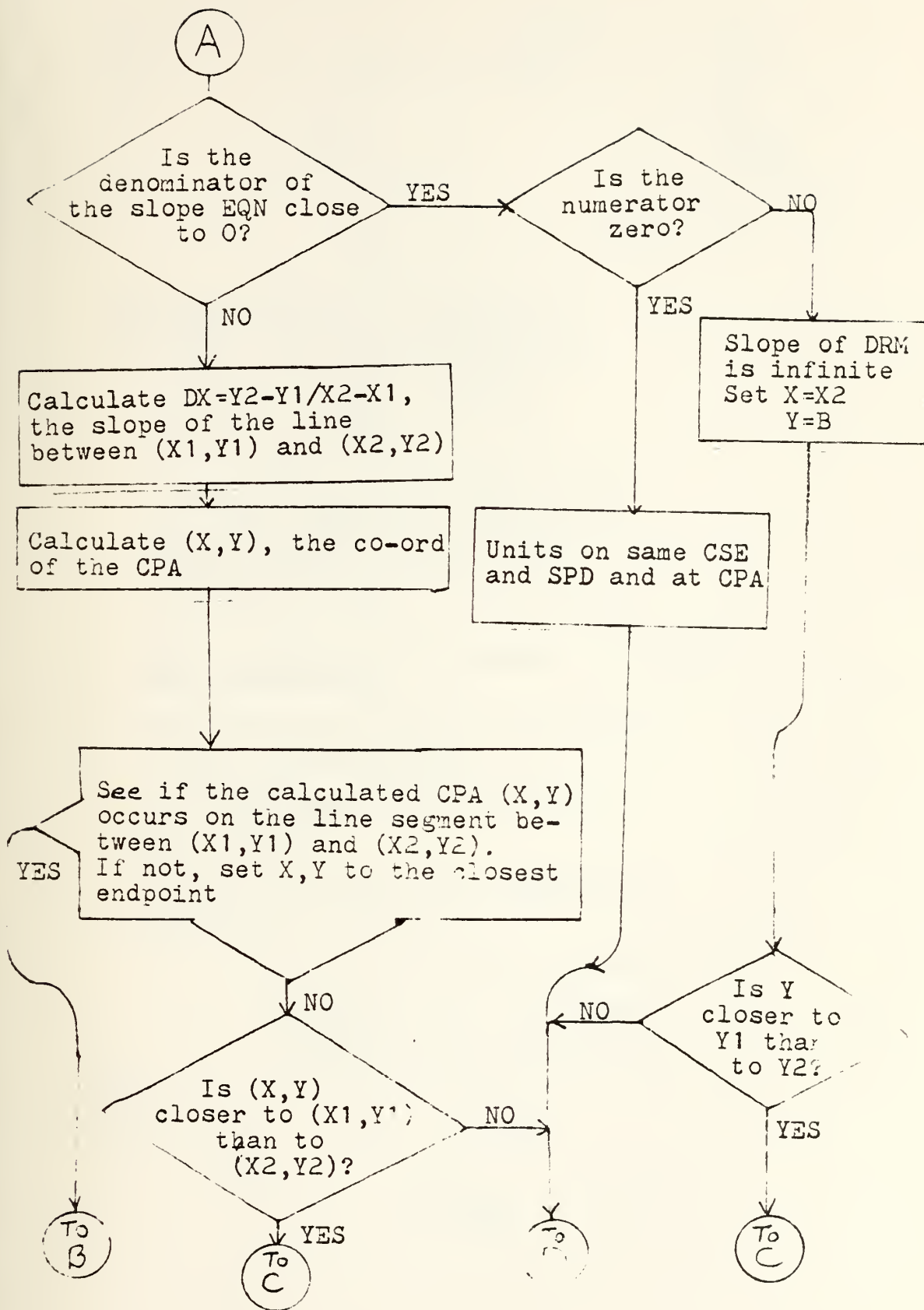
<u>Variable</u>	<u>Description</u>
A	Current X coordinate of the Ith unit.
B	Current Y coordinate of the Ith unit.
D1	Calculated distance the Nth unit moved this turn.
D2	Calculated distance the Ith unit moved this turn.
DX	Calculated slope of the DRM line.
ETA	Course of the Ith unit in radians.
I	Argument denoting the Ith unit.
NUNIT	Set to total number of enemy units.
RAD	Set to $\pi/180$. Used as degree to radian conversion factor.
T	Calculated time of this turn in hours.
TAU	Angle of the line from (A,B) to (X,Y) with the line from (X,Y) to (X,B).
THETA	Course of the Nth unit in radians.
TIME	Argument containing the elapsed time of this turn in minutes.
X	X coordinate of the calculated and final CPA.
X2	Current X coordinate of the Nth unit.
XDIST	Distance along the X-axis the Ith unit moved this turn.
Y	Y coordinate of the calculated and final CPA.
Y1	Y coordinate of the relative position of the Nth unit to the Ith at the beginning of this turn.

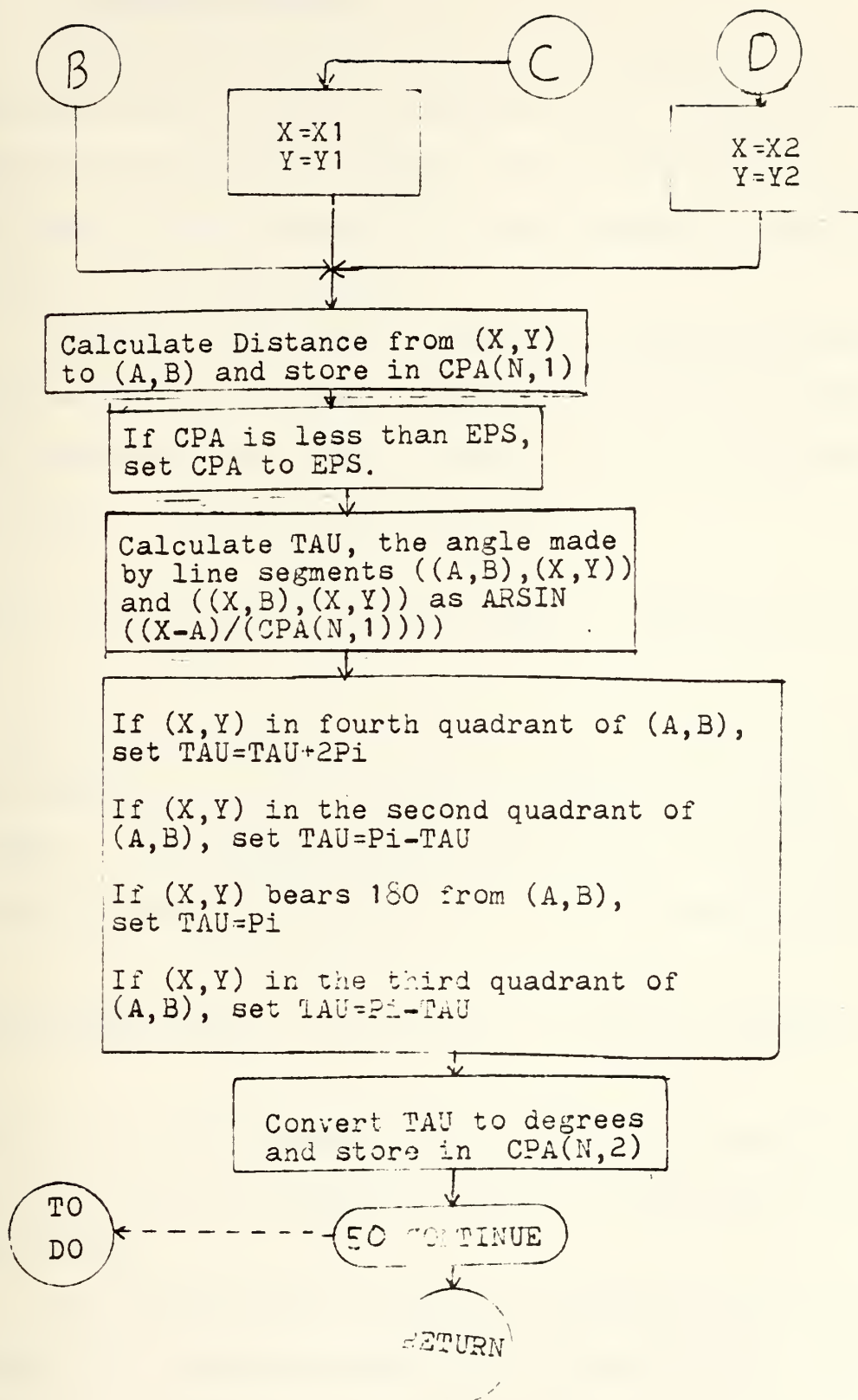
Table 14 (Continued)

<u>Variable</u>	<u>Description</u>
Y2	Current Y coordinate of the Nth unit.
YDIST	Distance along the Y-axis the Ith unit moved this turn.



Flowchart of Subroutine NLPCPA
Figure 16





11. Subroutine PSONAR

Subroutine PSONAR determines all possible passive sonar detections by the Ith friendly unit. Ships and submarines at any depth attempt passive detections if their sonar is off (passive) and their speed is less than 17 and 22 knots respectively.

There are two kinds of passive detections possible in this subroutine. First, an active sonar may be detected with a passive sonar. This is analogous to an ESM detection of a radar. Range is calculated as:

$$\text{Range} = 2 * (\text{Percent effectiveness}) * (\text{Range of active sonar})$$

If this is less than the calculated CPA, a detection occurs. Sonar ESM detections are stored in the last 10 columns of SONAR. An approximate range is provided to the detecting unit for an ESM passive detection. This is calculated as:

$$\text{Approximate Range} = \text{CPA Range} + \text{CPA Range} * (\text{DICE (N)} - .5)$$

DICE (N) is a U(0,1) random number. The approximate range will be uniformly distributed between one-half and 1.5 the range at CPA.

The second type of passive detection is the passive tracking of a target by sonar. The target's range, bearing, course, and speed are determined by the tracking unit. Any target, with the exception of aircraft, may be detected.

To calculate if a passive detection has occurred, the unit's maximum detection range is calculated as a function of target speed, searcher speed and where the target and searcher are with respect to the layer.

The curves in the top half of Figure 17 are used when a shallow sub is attempting to detect a unit on the surface or two submerged subs are attempting to detect one another at the same depth. This is called Case 1. The curves were approximated by a family of ellipses whose parameters are functions of target speed. The solid curves are those given in the SEATAG manual and the dashed curves those of the ellipse approximations.

The ellipse parameters for Case 1 are calculated as follows:

$$A = (\text{Target Speed}/5) + 15$$

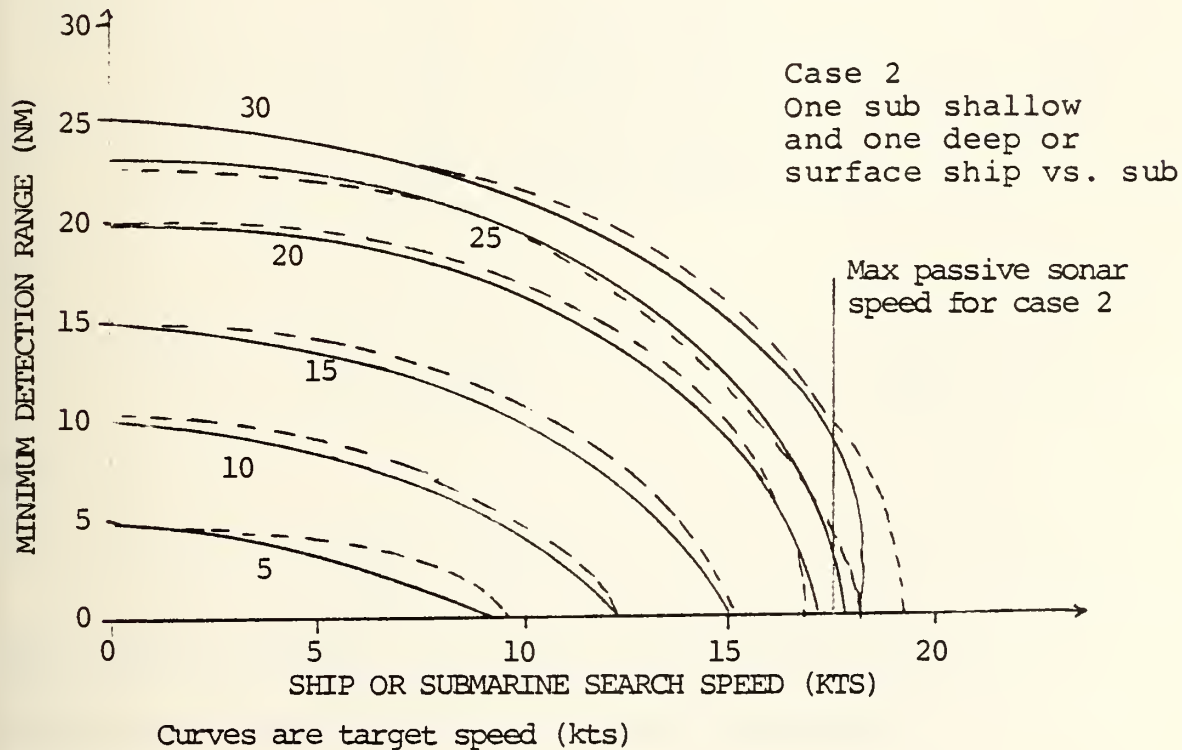
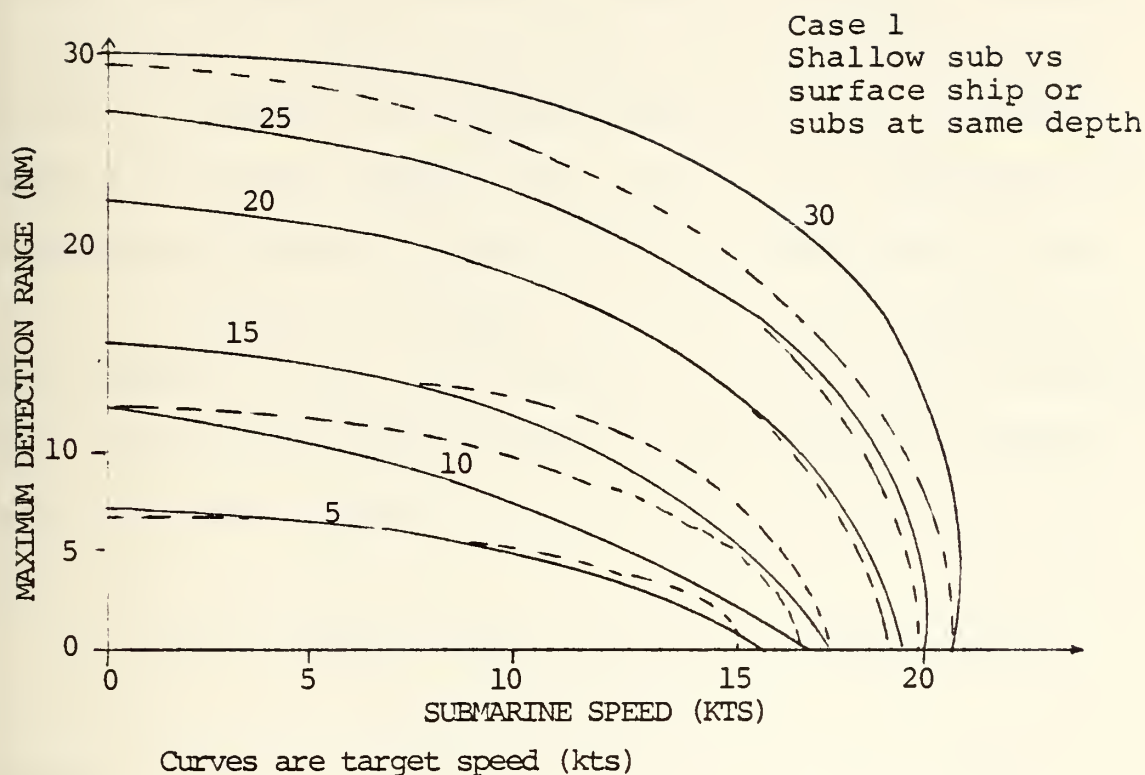
$$B = \text{Target Speed} + 2$$

$$X = \text{Searching unit speed}$$

For the ellipse

$$\frac{x^2}{A^2} + \frac{y^2}{B^2} = 1$$

Solving for $Y = B * (1 - (X^2/A^2))$ gives the maximum passive sonar detection range in n.m. calculated by the subroutine. The maximum range error is plus or minus 3 n.m. from the original curves.



Comparison of Passive Sonar Detection Curves

Figure 17

The curves in the bottom half of Figure 17 represent a cross layer detection situation. This is referred to as Case 2. They are used for any unit on the surface trying to detect any submerged submarine, for deep subs trying to detect surface units, and for two submerged submarines at different depths trying to detect each other.

A series of ellipses was also used to approximate the original curves. The values of X and Y remain unchanged. The A and B parameters for Case 2 are calculated differently as shown below:

$$A = \text{Target speed} + C * (15 - \text{Target speed})$$

where

$$C = .4 \quad \text{if} \quad \text{target speed} < 16$$

$$C = .6 \quad \text{if} \quad \text{target speed} > 15 \quad \text{and} \quad \leq 24$$

$$C = .7 \quad \text{if} \quad \text{target speed} > 25$$

and

$$B = \text{Target speed} \quad \text{if} \quad \text{target speed} < 21$$

or

$$B = (\text{Target speed}/2) + 10 \quad \text{if} \quad \text{target speed} \geq 21$$

The following table gives the A and B parameters as a function of target speed for 5 knot increments:

Case 2

<u>TGT SPD</u>	<u>A</u>	<u>B</u>
0	6	0
5	9	5
10	12	10
15	15	15
20	17	20
25	18	22.5
30	19.5	25

In Case (2) the vertical marks where the curves are truncated at a maximum of an 18 knot searcher speed. This in effect means much of the poorer fitting 30 knot curve is not used. The worst distortion is for a 17 knot search speed against a 25 knot target which results in a 7 n.m. range vice the 5 n.m. range obtained using the curves manually.

In some cases it is possible the searcher's speed will be greater than the calculated A values. In this case no detection should be possible so A is set equal to zero, insuring Y will be zero. If the calculated Y value is greater than the listed sonar max range on the characteristic card, Y is set to the maximum range from the characteristic card. This happens most frequently with surface ship short range sonars.

Once Y is determined, it is multiplied by the percent of maximum range (the PRMAX value) to give the final range. This is then compared with the CPA to determine if

a detection did in fact take place. Detections are then stored sequentially by row in the first ten columns of array SONAR.

The argument I passed to PSONAR specifies the row of the friendly unit attempting to make the passive detections and is used in PSONAR to access information for the Ith unit for array DATA. The argument IPLAT designates the searching unit as a surface ship or submarine.

Subroutine ASONAR is called from subroutine DETECT. Subroutine NLPCPA must have been previously called last for unit I. Values from array PRMAX with the third subscript $K = 5$ are used for the percent of maximum range.

Table 15 provides a list of local variables and Figure 18 a flowchart for subroutine PSONAR.

Table 15

Subroutine PSONAR, Local Variables

<u>Variable</u>	<u>Description</u>
A	Parameter of ellipse in standard form.
B	Parameter of ellipse in standard form.
B2	Used to calculate B as a function of target speed.
C	Used to calculate A as a function of target speed.
I	Argument passed from DETECT denoting the Ith unit searching passively.
IA	Column subscript in do loop 33 used to transfer data on the Mth detection into array SONAR.
II, JJ	Row and column subscripts for printing out columns of array SONAR.
IK	Column subscript used in do loop 55 to transfer the MMth passive detection data into array SONAR.
IPLAT	Argument passed from DETECT denoting the type of searching platform, 1 for a submarine, 2 for a surface unit.
M	Counter for the number of passive sonar ESM detections.
M1	Lower bound on the row subscript of array SONAR when printing out passive detections.
MM	Counter for the number of passive sonar detections. Starts from M.
N	Row subscript denoting the Nth enemy unit used in do loop 35 and 60.
NTGT	Last three digits of the Nth unit's type number.
NUNIT	Set to total number of enemy units.
RANGE	Calculated final detection range in n.m.

Table 15 (Continued)

<u>Variable</u>	<u>Description</u>
X	Searchers speed (Ith units speed in knots).
Y	Calculated detection range before modification by PRMAX.

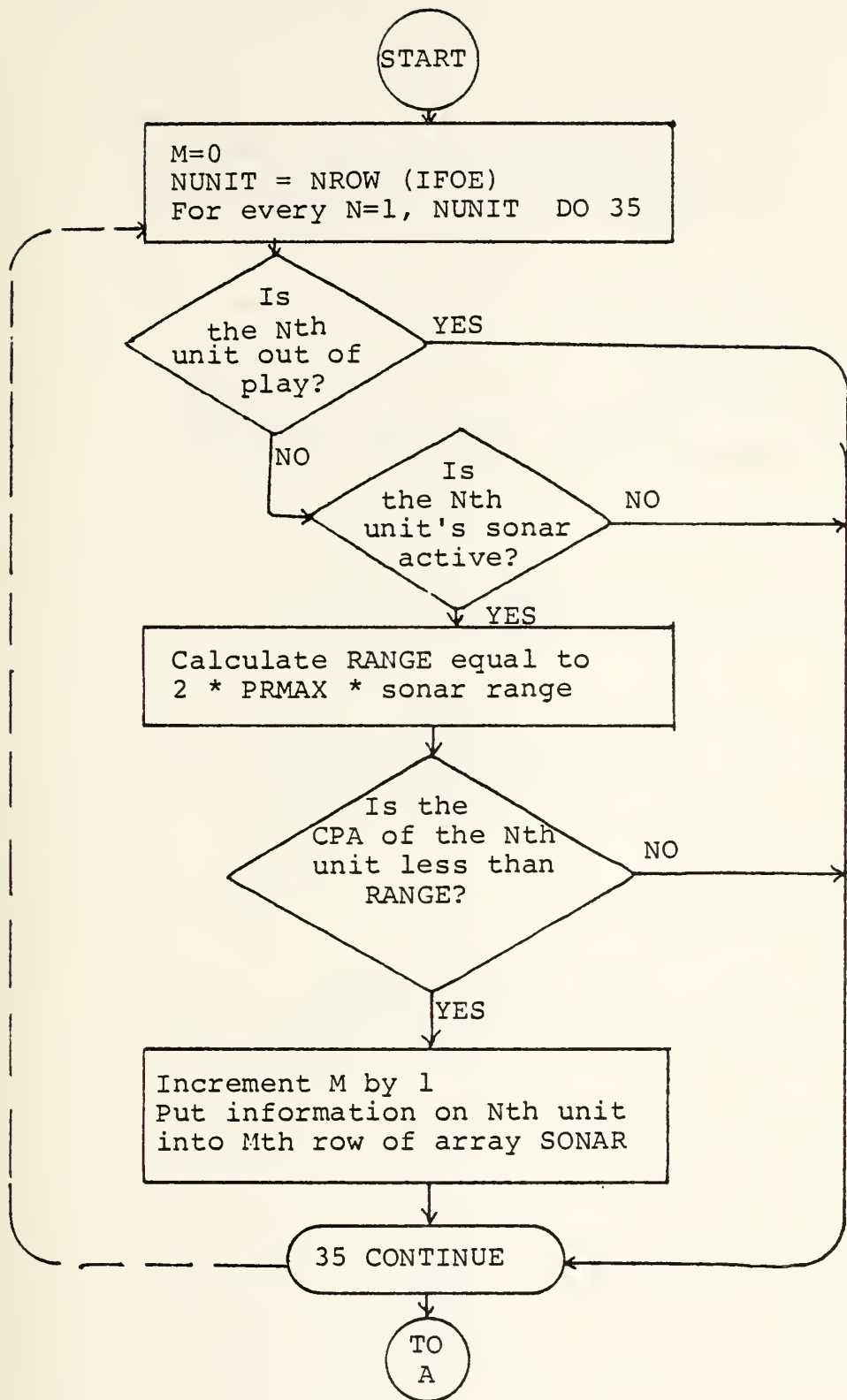
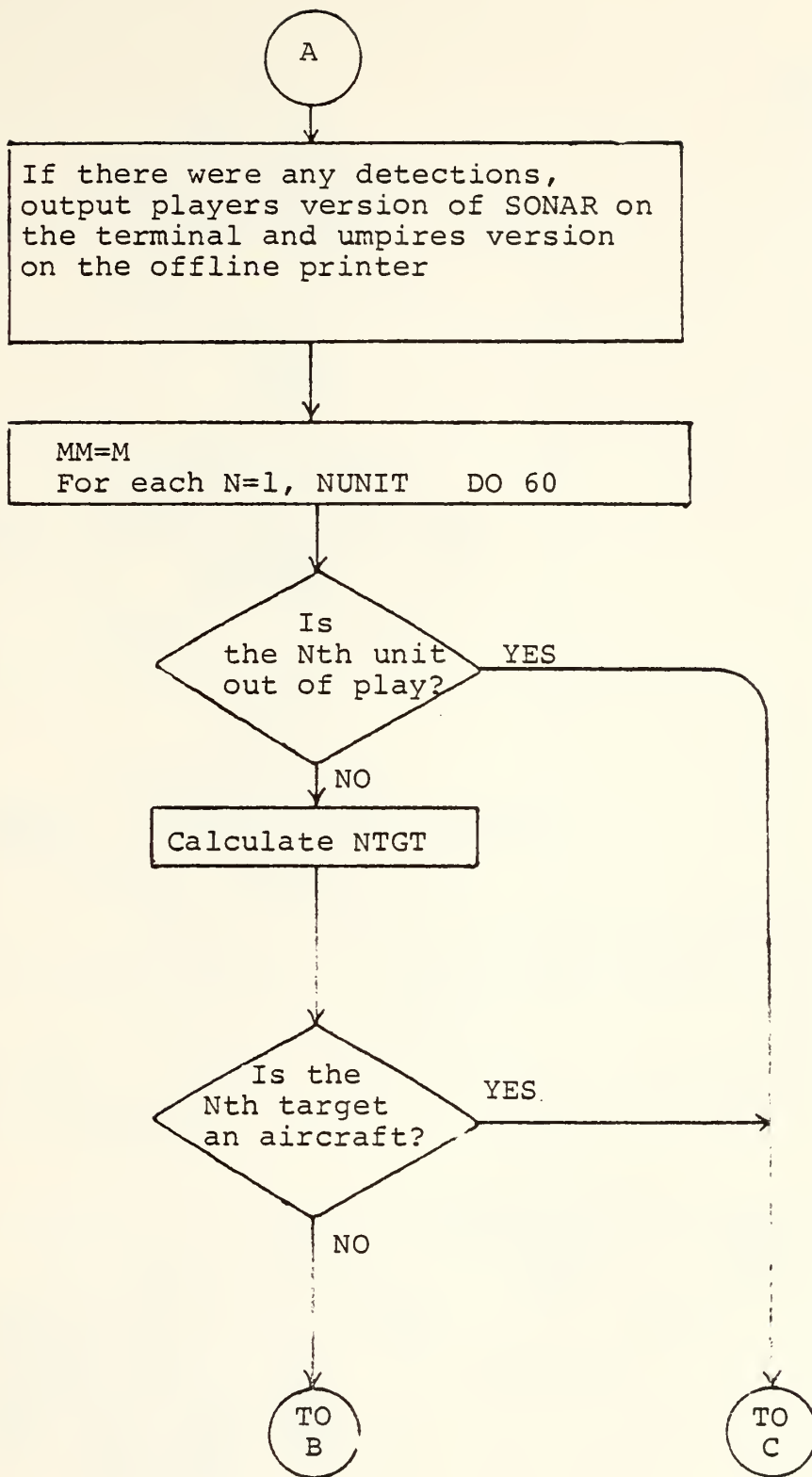
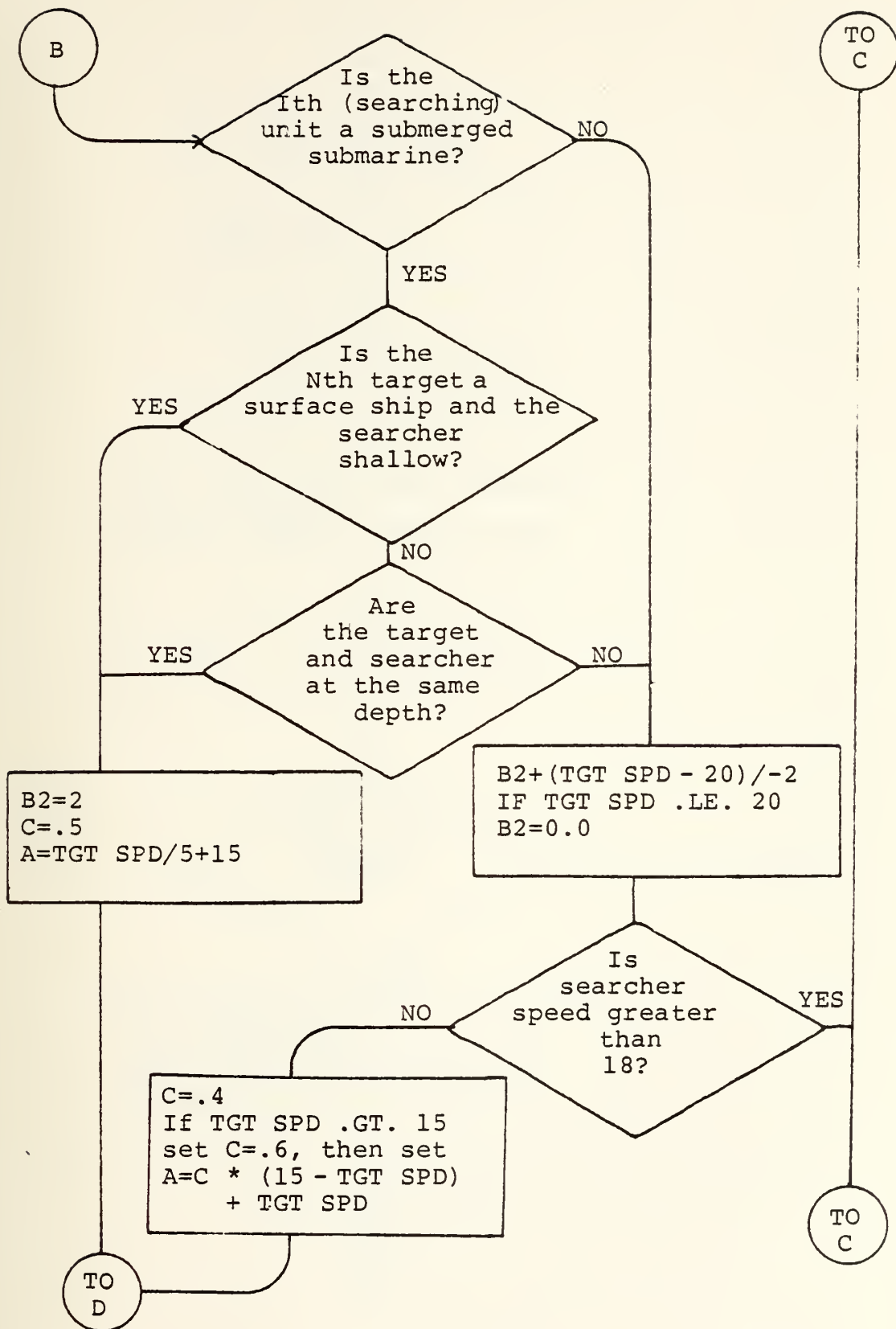
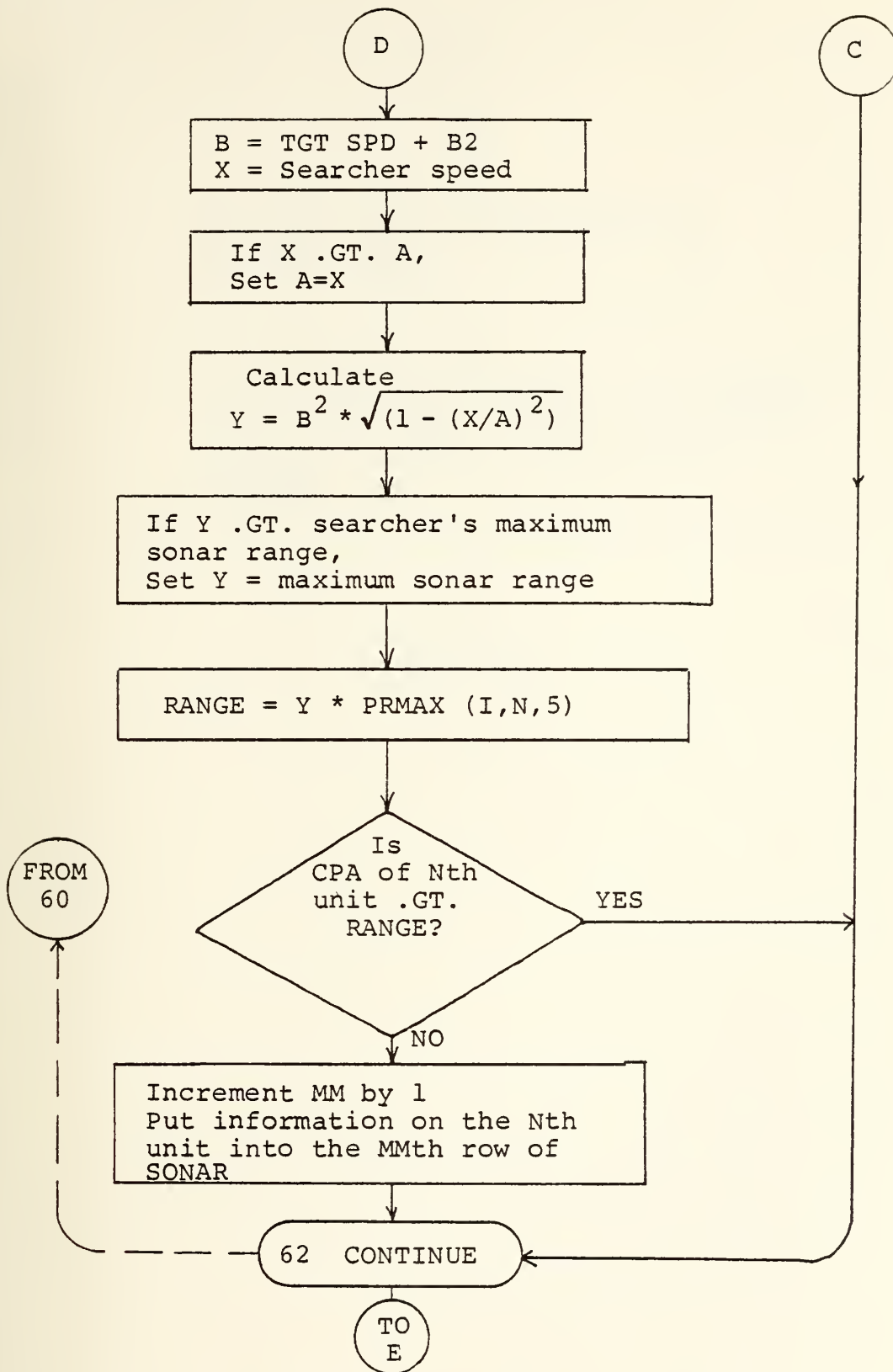


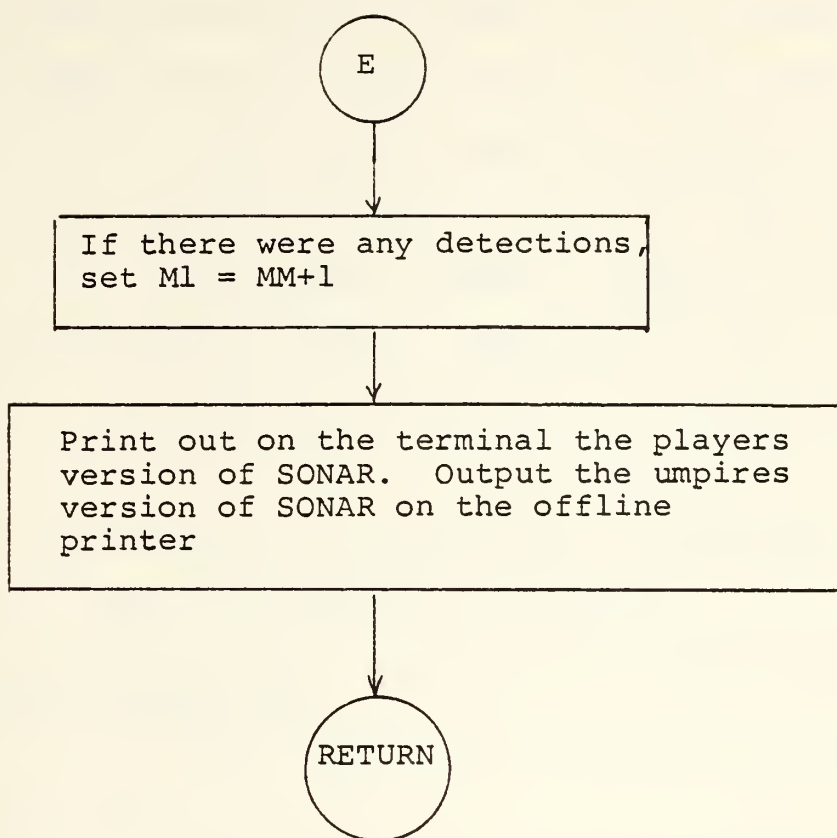
Figure 18

Flowchart of Subroutine PSONAR









G. DESCRIPTION OF FORTRAN PROGRAM CHANGE

CHANGE allows the player to change a unit's course, altitude or depth and turn radars, sonar, and HF communications on or off. Units may be brought in and out of play by changing their ID. Units brought into play may have their X and Y coordinates changed to insure they enter play at an appropriate location. Units out of play are set to (0,0). CHANGE is executed once per game turn by CASTAG EXEC.

Integer inputs that are incorrect for a particular parameter of a unit are corrected as follows:

1. Speeds greater than listed maximum speed are set to unit's max speed.
2. An altitude other than 0 for a surface unit will be set to 0.
3. Any course greater than 360 will be set to 360.

A unit which has had a non-existent sensor activated will not affect the detection routines. The maximum range for that sensor will be zero in the data base resulting in a consistent zero detection range.

ID numbers and X or Y coordinates are not screened in any way and must be corrected by cycling that particular unit back through the change procedure. Figure 19 is a flowchart of CHANGE and Table 16 is a list of the variables used in CHANGE.

Table 16

List of Variables and Arrays for Program CHANGE

<u>Variable/Array</u>	<u>Description</u>
DATA (50, 20)	Integer array holding either Red or Blue data in the same format as in the data base.
NCOL	Number of columns being used for data in array DATA. Set to 15.
NREAD	Used to designate the terminal as the read device for FORTRAN READ statements. Set to 5.
NWRITE	Used to designate the terminal as the write device for FORTRAN WRITE statements. Set to 6.
NROW	Maximum number of units (rows) array DATA will hold. Set to 50.
ZTIME	Game clock time in minutes. Printed out for reference.
IDENT	Input by user. 1 designates Red; 2 designates Blue.
IDISK	DSRN designating which file to access for information. Set to IDENT + 2, giving 3 for Red, 4 for Blue.
N	Input by user telling the program what column of array DATA will be changed.
ITYPE	Last three digits of Nth units type number.
IDNO	Row subscript of DATA designating that unit will have its ID number changed.
I	Used throughout the program as a variable row subscript for array DATA.
J	Used throughout the program as a variable column subscript for array DATA.

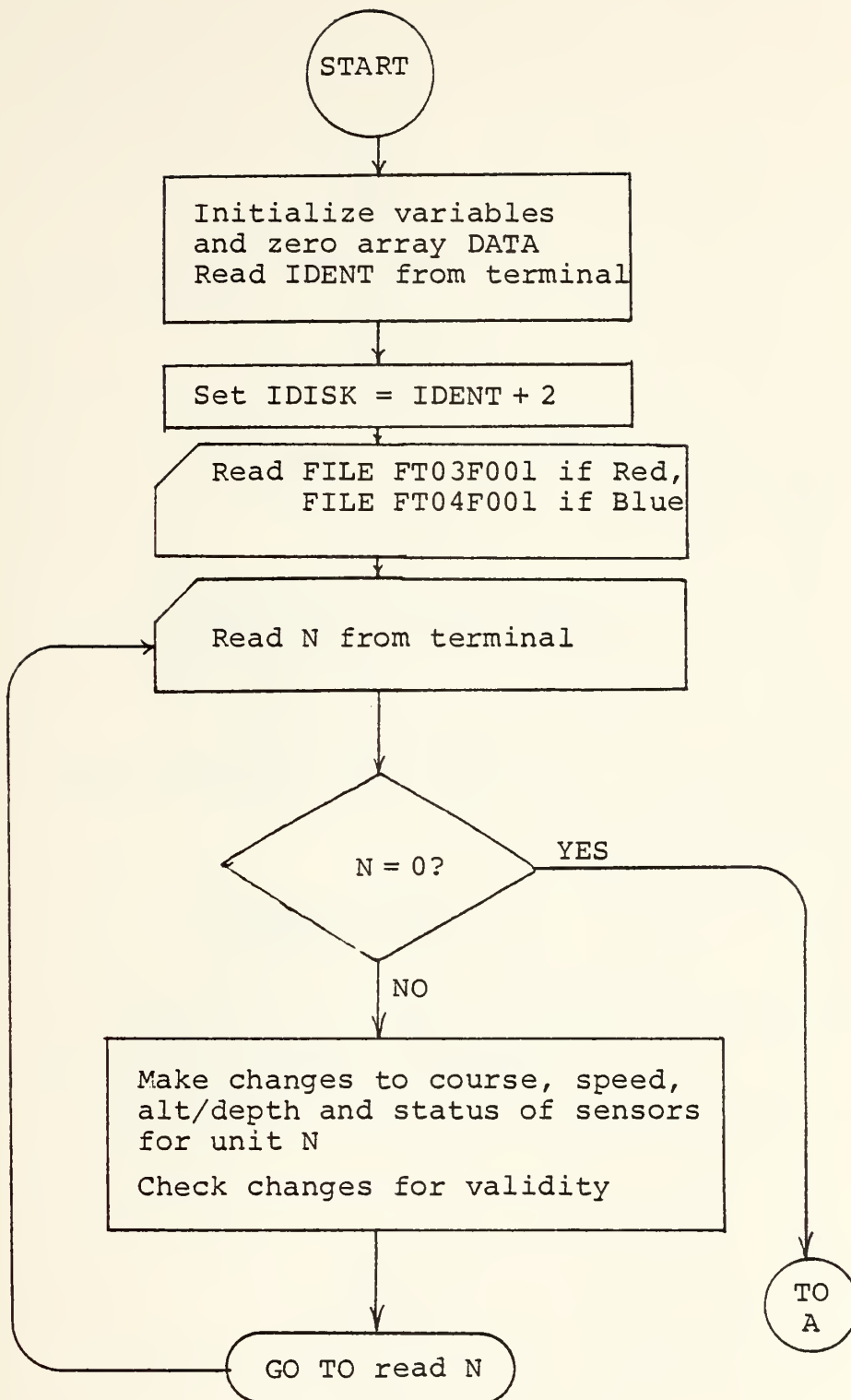
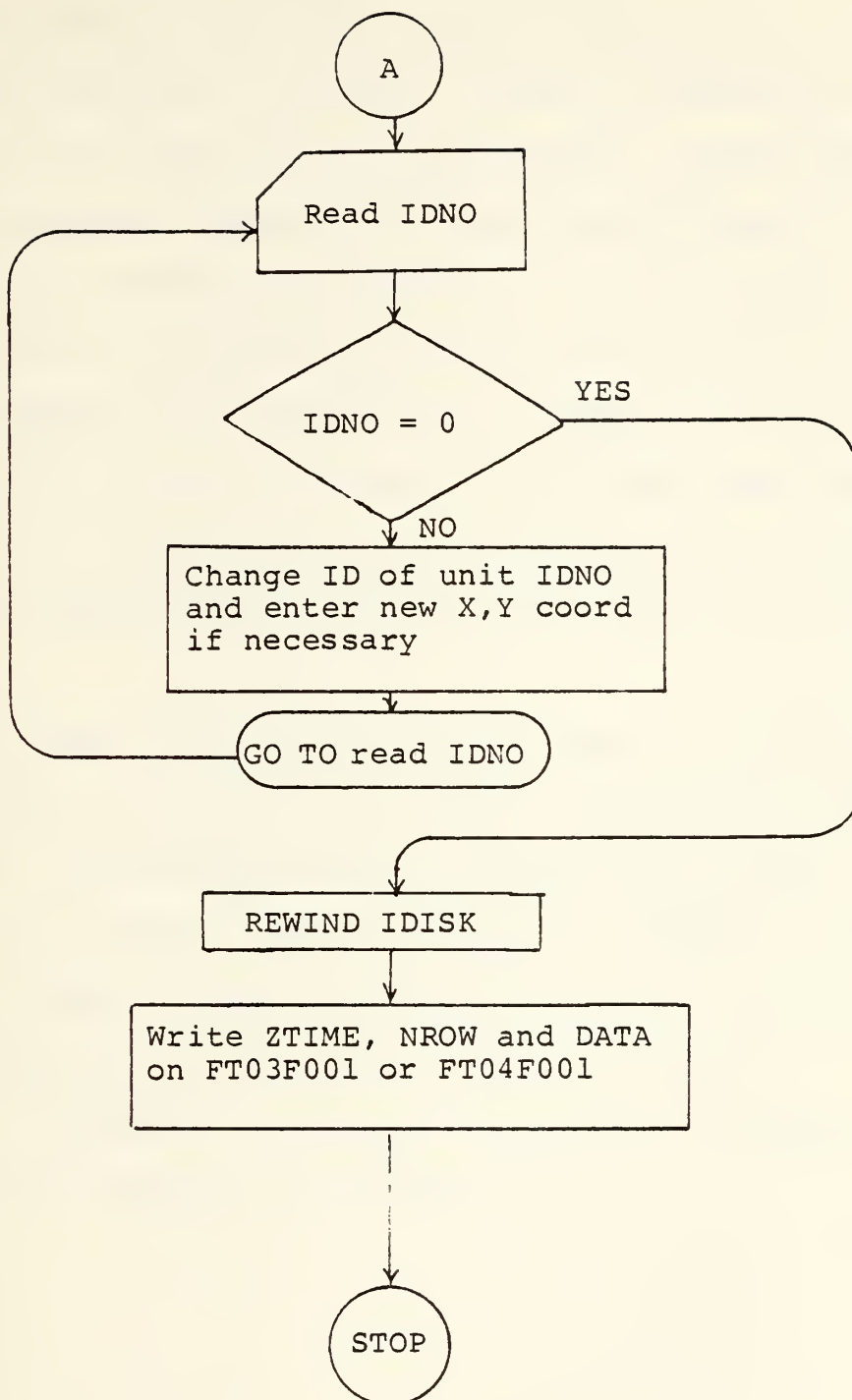


Figure 19

Flowchart of FORTRAN Program CHANGE



H. DESCRIPTION OF FORTRAN PROGRAM UPDATE

UPDATE combines two files into the new data base prior to the next execution of SEARCH. UPDATE is executed once at the end of the game turn by CASTAG EXEC. UPDATE READS the two files which contain the user entered changes to the data base (FT03F001 and FT04F001) into two arrays. File FT02F001 is prepared to be written on from the top by a REWIND command which destroys the existing information. File FT02F001 was the data base which had been saved up to this point to provide a backup if either of the two new files were inadvertantly destroyed during execution of CHANGE.

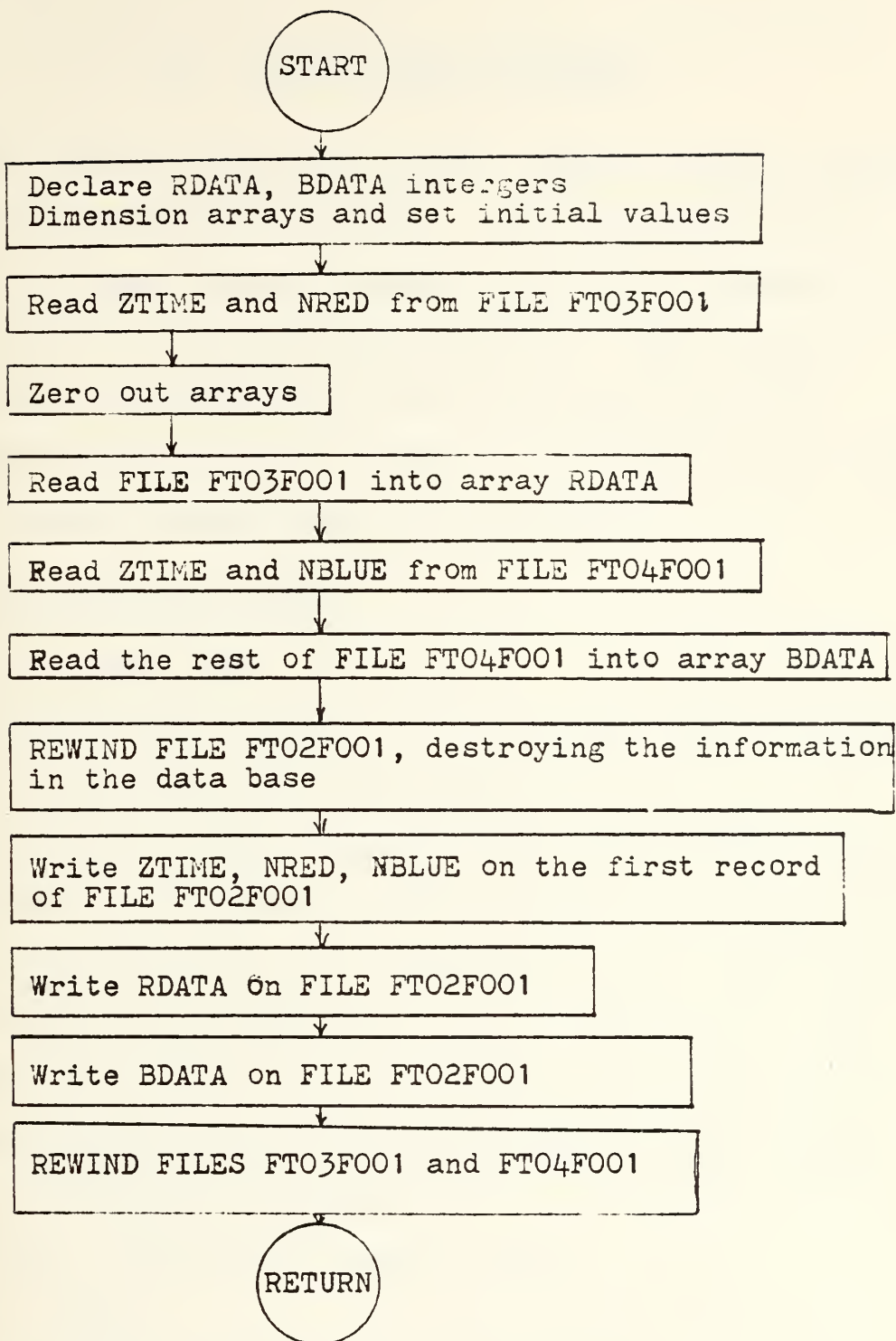
Arrays RDATA and BDATA are written back on FT02F001 in the same format as the old data base. FT02F001 and FT04F001 are rewound. Note that should the users have entered different turn lengths during execution of SEARCH at the terminals, ZTIME will be fixed for next turn at both terminals at whatever was entered by the Blue player.

Table 17 lists the variables and arrays in UPDATE and Figure 20 is a flowchart of UPDATE.

Table 17

List of Variables and Arrays for Program UPDATE

NCOL	Number of columns being used in the data structure. Set to 15.
NROW	Maximum number of units (rows) per side. Set to 50.
NRED, NBLUE	Total number of RED and BLUE units, respectively.
ZTIME	Total game time in minutes.
NDISK	DSRN for the data base. Set to 2 designating file FT02F001 for READ and WRITE statements.
RDISK, BDISK	Integers designating FT03 and FT04 as READ files.
RDATA (50, 20)	RED half of the original data base which has had the changes made.
BDATA (50, 20)	BLUE half of the original data base which has had the changes made.
I	Row subscript for RDATA and BDATA.
J	Column subscript for RDATA and BDATA.



Flowchart of FORTRAN Program UPDATE

Figure 20

IV. USER'S GUIDE TO CASTAG

This chapter discusses terminal inputs, program outputs, and program recovery procedures. Some familiarity with SEATAG is assumed as well as some knowledge of FORTRAN format "I" and "F" input type declarations. The line and letter delete instructions refer specifically to the DECWRITER T-300 terminals in the NPS Computer Center. Other terminals should have analogous functions.

A. CASTAG TERMINAL INPUTS

The terminals have two keys which can be used to correct input errors before returning the carriage. The "@" symbol is used to delete the previous character in an input line. More than one "@" can be used per line. For example, entering a format lI2 parameter, "lA" is typed. To correct the illegal character "A", type "@2" resulting in "lA@2" which will be read as "l2" following the carriage return. The "]" symbol typed at the end of a line will cause the terminal to delete the line following the carriage return allowing the user to re-enter the line.

To begin execution of CASTAG, enter:

```
$ CASTAG
```

the terminal will print:

```
THIS IS THE SEATAG EXEC ROUTINE.  
THIS PROGRAM WILL REQUIRE A DATA BASE  
WHICH HOLDS ALL THE APPROPRIATE SHIP AND  
AIRCRAFT CHARACTERISTICS.
```


ENTER THE USER NUMBER OF THE OTHER PLAY INCLUDING P OR G FOR USER TYPE, MAKING A TOTAL OF 5 DIGITS, EG; "1059P".

>

The ">" symbol means the terminal is waiting for an input to CASTAG EXEC. An illegal input will cycle an EXEC program back to the input instruction. In this example, the user entered "1059P". The terminal replies with:

IS OTHER PLAYERS ID = 1059P ? ENTER Y OR N.

>Y

Answering with "N" would give the user a chance to re-enter the user ID. Following a "Y" entry, the terminal prints:

FILE FT02F001 IS THE DATA BASE. THIS FILE MUST BE SENT TO THE CONSOLE OF THE OTHER PLAYER IF NO COPY EXISTS IN HIS FILES. IF YOU ARE THE OTHER PLAYER, YOU WILL HAVE TO READ FILE FT02F001.
IF NO TRANSFER OF DATA BASE REQUIRED ENTER N.
TO TRANSFER DATA BASE TO OTHER PLAYER ENTER "T".
TO READ THE DATA BASE FROM THE OTHER PLAYER ENTER "R".
>R

The example shows the user wants to read the data base from the other terminal.

FILES:- NO RDR, NO PRT, NO PUN
IF YOU HAVE A FILE SPOOLED WAITING TO BE READ, ENTER "Y". IF NOT ENTER ANY OTHER LETTER EXCEPT N. THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE FILE IS AVAILABLE OR UNTIL Y OR N IS ENTERED. TO GO AHEAD WITHOUT READING THE FILE ENTER N.

The message shows no files waiting to be read. Consequently, an entry of any letter except "Y" or "N" should be made. Either or both of the messages will indicate that there is a spooled file waiting. These are:

** CARDS XFERED BY 1059P05 **

or

FILES:- 01 RDR, NO PRT, NO PUN

Assuming the user had entered "N", and then received one of the two messages above, the terminal repeats:

```
IF YOU HAVE A FILE SPOOLED WAITING TO BE READ,  
ENTER "Y". IF NOT ENTER ANY OTHER LETTER EXCEPT N.  
THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE  
FILE IS AVAILABLE OR UNTIL Y OR N IS ENTERED. TO  
GO AHEAD WITHOUT READING THE FILE ENTER N.  
>y
```

Entry of "Y" will output a verification of the file read:

```
OFFLINE READ      FILE      FT02F001 P1
```

An entry of "T" will print one message:

```
** CARDS XFERED TO 0000G **
```

and skip the preceding sequence.

Entry of "N" at any point skips the file transfer section and proceeds with program execution, even if there is no data base at one or both terminals.

The next line, "EXECUTION BEGINS. . ." indicates the start of a FORTRAN program. In this case it is program SEARCH. All inputs to FORTRAN programs will have format instructions for the required entry. All are integer entries with the exception of one F4.0 entry. Integer formats are specified by "FORMAT I4" or "I12" or "I11" or some combination thereof. Any illegal entries will cause either program termination or a "fix-up" which will probably cause the input to be read as zeros.

SEARCH requires three entries. The first is the random number seed, of up to seven integers. A zero may be entered

in which case the program sets the random number seeds. Different random number seeds should be entered at the two terminals. The next entry is the duration of the next turn in minutes. Both terminals should enter the same value. The maximum value is 1440 minutes or 24 hours and the minimum is zero. Only the time specified by the Control Group should be entered. Last, a single digit entry, "1" or "2", tells the program whether it is Red or Blue. Following these entries, a printout of detection information for each side is printed out at the terminal. Interpretation of this output is covered in Section B of this chapter. Below is an example of the inputs to SEARCH:

```
EXECUTION BEGINS...
ENTER THE RANDOM NUMBER SEED, FORMAT I7.

>1219503
ENTER THE TIME, IN MINUTES, OF THIS TURN, F4.0.
>60
ENTER "1" IF YOU ARE THE RED PLAYER, "2" IF BLUE.
FORMAT I11.

>2
```

The entries show a seed of "1219503", a sixty minutes turn length and that this terminal is Blue this turn.

Conclusion of program SEARCH and the start of the next FORTRAN program is indicated by

```
"EXECUTION BEINGS...
ENTER 1 FOR RED, 2 FOR BLUE, FORMAT I11".
```

The next FORTRAN program, CHANGE, has begun execution. This is an interactive sequence that allows the user to change parameters and bring units in and out of play. The initial access is by row, so the first unit in the data base is

unit "1". There can be up to 50 units so these entries are made in I2 format. A code number (2 through 8) is input to determine which value is to be changed followed by the new value when requested. All changes to a unit should be made first before moving on to another unit although it is possible to cycle through the changes as many times as required. Appendix B provides an example of an actual CHANGE execution.

Following entry of the last change to the last unit, program control is returned to CASTAG EXEC. The next sequence transfers the files altered by CHANGE.

```
NOW EACH PLAYER SENDS HIS UPDATED FILE TO THE OTHER
  PLAYER. TO INITIATE TRANSFER,
ENTER "R" IF YOU ARE RED, "B" IF BLUE.
>b
YOU ENTERED B . IS THIS CORRECT? ENTER "Y" OR "N".
>w
```

The above sequence tells the CASTAG program if it is Red or Blue.

Two messages will indicate a file waiting to be read as in the data base transfer.

```
** CARDS  XFERED BY 1059P05 **
** CARDS  XFERED TO 1059P   **
FILES:- 01 RDR, NO PRT, NO PUN
DO YOU HAVE A FILE SPOOLED TO BE READ? ENTER "Y" OR "N".
>y
OFFLINE READ      FILE      FT03F001 P1
```

The XFERED BY" message may appear while still executing CHANGE. This means the other terminal user has finished his changes and initiated transfer. The other player still must wait to read your change file before continuing. The rest of the

sequence is carried out as if "R" was entered in the transfer at the beginning of the program.

After both transfers have been completed, a list of files is output for comparison to four filetypes. Each terminal should have all four before proceeding to the next turn.

FILENAME	FILETYPE	MODE	NO.REC.	DATE
FILE	FT01F001	P1	10	2/19
FILE	FT02F001	P1	2	2/19
FILE	FT04F001	P1	1	2/19
FILE	FT03F001	P1	1	2/19

YOU SHOULD HAVE FILE TYPES FT02F001, FT03F001, FT04F001
AND FT01F001 ON YOUR DISK AT THIS TIME.

CASTAG executes UPDATE and prints:

TO PLAY ANOTHER TURN, ENTER Y. TO QUIT ENTER N.

This entry lets the user go back to SEARCH to play another turn or to terminate the program.

B. DECODING/ENCODING TYPE NUMBERS

Each unit in CASTAG is given a four digit code for identification as to the unit's side, platform, kind and class. These numbers are called type numbers and are assigned when the data base is created. They are decoded by the participants to determine information about specific detections. Table 18 provides definitions for the first three digits. Table 19 gives the fourth digit assignment for all the units currently available in CASTAG.

Table 18

First Three Digits of Type Number Assignment

DIGIT	NAME	VALUES	DEFINITION
First	Side	0	Neutral
		1	Red
		2	Blue
Second	Platform	1	Submarine
		2	Aircraft/helicopter
		3	Surface vessel
Third	Kind	0	Merchant or civilian
		1	Submarine
		2	Aircraft carrier
		3	Large surface combatant (CG,CLG,DDG)
		4	Small surface combatant (FF,FFG, PG)
		5	Service force/ amphibious force (AOE)
		6	Fighter, Interceptor, Attack Aircraft
		7	ASW fixed wing air- craft (S-3,P-3)
		8	Bomber, Recon or AEW aircraft
		9	Helicopter

Table 19

Type Numbers, Fourth Digit

BLUE UNITS	NUMBER	CLASS
Submarines	1	Los Angeles
	2	Sturgeon
Aircraft carrier	1	USS America
CG, CLG, DDG, DD (Largest surface combatants)	1	Leahy
	2	C. F. Adams
	3	Coontz
	4	Barry
	5	Spruance
FF, FFG, PG, PGM (Small surface combatants)	1	Perry
	2	Brooke
	3	Knox
	4	Garcia
Service force/Amphibious	1	Sacramento
Fighter/Attack aircraft	1	KA-6D Tanker
	2	EA-6B ECM support
	3	A-6 Corsair
	4	A-7 Intruder
	5	Phantom II
ASW fixed wing aircraft	1	S-3A
	2	P-3C
Bomber, Recon, AEW aircraft	1	E-2C
Helicopters	1	SH-3H
	2	SH-2D

Table 19 (Continued)

RED UNITS	NUMBER	CLASS
Submarines	1	Charlie SSGN
	2	Victor SSN
	3	Echo II SSGN
	4	November SSN
	5	Juliett SSG
	6	Foxtrot SS
Aircraft carriers	1	Kiev
	2	Moskva
CG, CLG, DDG, DD	1	Kara
	2	Kresta I Kynda
	3	Kresta II
	4	Krivak
FF, FFG, PG, PGM	1	Nanuchka
Service/Amphibious force	1	Chilikin (AOE)
Fighter/Attack aircraft	1	MIG 23
	2	YAK 36 Forger
ASW fixed wing		None
Bomber/Recon aircraft	1	Bear
	2	Badger

C. CASTAG TERMINAL OUTPUT

Following execution of SEARCH, the detection information for that side is output on the terminal. If there were no detections by a particular unit or no detections of a certain type there is no output. It is possible for there to be no detections, and hence no output, at all. All detections, with the exception of active sonar, are of opposing units.

The first output is surveillance information available only at 0001, 0601, 1201 and 1801 game time. The information is presented as follows:

UNIT TYPE	X-COORD	Y-COORD	PLATFORM	COURSE
21	535	476	0	0
23	504	491	47	95

Unit type is the first two digits for the type number. In the example above, "21" is a Blue submarine and "23" is a Blue surface unit. The (X,Y) position is the current one. The platform is the last two digits of the type number. "0" indicates the information is not available. The course, if provided, is rounded to within 10 degrees of the true course.

Any HF detections are output next in the following format:

HF DETECTIONS OF ENEMY COMMS	
X COORD	Y COORD
510	510

This example informs the user there was an enemy HF transmission originating from (510, 510) during this turn.

Each unit, in row order, will now have all of its detections printed. The output order varies with the platform as shown below:

ORDER	SUBMARINE	SURFACE	AIRCRAFT
1	Passive Sonar	ESM	Visual
2	ESM	Visual	ESM
3	Radar	Radar	Radar
4	Visual	Passive Sonar	Active Sonar
5	Active Sonar	Active Sonar	N/A

Passive sonar detections have the following format:

```
DETECTIONS OF ACTIVE SONARS BY UNIT 111
  DEPTH   BEARING   APPROX RANGE
    0       89       10
```

```
PASSIVE DETECTIONS BY UNIT1011
  COURSE  SPEED   DEPTH   RANGE  BEARING
    160     30     0       12     89
    100     7      1        2    199
```

Course, speed, and depth are as per the information in the data base. The bearing is in both cases the bearing at CPA, not necessarily the most current bearing. The passive detection range is the range at CPA. The approximate range for ESM detections of active sonars is a uniformly distributed random variable with the mean at the range at CPA. These detections are by unit ID number 111.

All ESM detections appear in the following format:

```
ESM INTERCEPTS BY UNIT 333
  RADAR   BEARING   PLATFORM
    13     91       33
    12     26       29
RADAR TYPE 12 = SURFACE SEARCH TYPE 13 = AIR SEARCH
```


In the above example unit 333 has detected an air search radar (type 13) and a surface search radar. Both bearings are the radar bearings at CPA. The platform number is the middle two digits of the type number. "33" represents a Red CG/CLG and "29" is a Red helo.

Air and surface search radar detections appear jointly by unit. In the following example, unit 666 has detected two targets. The "12" designates the S/S radar and "13" the A/S radar. Notice that although the air target has a speed of 10 knots it must still be an air target of some type. It could be a helicopter or an aircraft presumed to be orbiting. "COMP" is the number of targets in the unit or its composition. All detections are single targets so this information is superfluous.

RADAR DETECTIONS BY UNIT 666						
	RADAR TYPE	COURSE	X COORD	Y COORD	SPEED	COMP.
	12	160	547	448	10	1
	13	225	510	510	10	1

Active sonar detections may be either friendly or enemy submarines. An example of a detection printout is provided below.

ACTIVE SONAR DETECTIONS BY UNIT 50				
DEPTH	COURSE	X COORD	Y COORD	SPEED
1	95	535	473	10
1	100	534	472	7

Unit 50 has detected two submarines in close proximity. A depth of "1" indicates they are both shallow while a "2" would indicate deep. (X,Y) position, course and speed are as previously explained.

Visual identifications, although only occurring at short range, provide reasonably complete target identification. In the example provided, unit 333 has two detections. Complete type information identifies "1332" as a Red CG of the KARA class. "1293" is a KA-25 Hormone helicopter. The altitude in the helicopter's case is 1000 feet. Bearing is still the bearing at CPA. The rest of the information is self explanatory.

VISUAL DETECTIONS BY UNIT 333

TYPE	ALT	COURSE	X COORD	Y COORD	BEARING	COMP
1332	0	245	419	479	355	1
1293	1	255	497	476	244	1

An example of the terminal output for every kind of detection has been provided. This information is in fact what the players will be basing their tactical decisions on. Section D provides a similar discussion of the offline printer output.

D. CASTAG OFFLINE PRINTER OUTPUT

Each terminal prints five sections in order. Each section is usually no more than a page in length. The contents and use of each section is discussed in order of appearance.

A representative from the Control Group should collect the printouts and distribute sections two and three to their respective sides. Both copies of Sections one, four, and five should go to the Control Group.

1. Umpire Summary of Detections

This output matches that of each terminal with the same detections in the same order as the originating terminal. The Control Group will need one from each terminal to get a complete picture of the preceding turn. All the information on the terminal printout is present with the addition of the ID number of the unit detected, the range and bearing of CPA and other pertinent information. The CPA range is especially important for playing engagements as this will be the range for all firing calculations. An example of the information provided in this printout is reproduced below.

```
UMPIRE COPY, PASSIVE SONAR DETECTIONS, UNIT 51
  ID SPD DEPTH  CSE  X    Y    TYPE  RGE  BRG
  555   17    0    90  548  491 2347    5   60
```

```
UMPIRE VERSION OF VISUAL SIGHTINGS, UNIT 71
  TYPE    ALT.  COURSE X COORD Y COORD BRG  COMP  CPA  RNG  ID
  2333     0    90      513   480  67    1    11   333
```

ESMDET, UMPIRE VERSION ID = 71

```
RADAR  BEARING  PLATFORM ID NO.  CPA RANGE
```

```
  12      88      32      444      25
  13      88      32      444      25
  13      70      34      555      45
  13     101      34      666      71
```

RADAR TYPE 12 = SURFACE SEARCH TYPE 13 = AIR SEARCH

```
UMPIRE VERSION OF RADAR DETECTIONS, UNIT 71
```

```
  ID  TYPE RADAR  CSE  X GRID Y GRID SPEED COMP CPA RGE & BRG
  333 2333    12   90  513  480   11    1    11   67
```

2. Red/Blue Cartesian Plot

For each terminal, a plop of all the side's forces in a 1000 by 1000 mile cartesian plot in the positive quadrant is provided. All detections which specified an (X,Y) position

are also plotted on this page. The two sides' units are plotted with different symbols ("+" and "."). This is distributed to the players.

3. Status of Forces Table

Each side receives page 3 which lists their forces and the current game time in hours and minutes. An example is provided below:

STATUS OF FORCES AS OF 8 HRS. 30 MIN											
UNIT ID	SPEED KTS	DEPTH /HT	COURSE DEG T	X GRID	Y GRID	TYPE	MAX SPD	MAX S/S	MAX A/S	MAX SNR	
50	30	0	265	476	478	1332	34	25	150	10	
51	10	0	265	530	488	1333	35	20	150	10	
52	10	0	265	532	448	1346	34	20	150	10	
60	7	1	100	555	467	1110	30	100	0	30	
0	0	20	90	900	500	1281	450	200	0	0	
71	10	1	225	493	472	1293	120	30	30	2	

The last four columns are the maximum speed and sensor ranges for the unit in question. Note that the fifth is out of play.

4. Umpire's Cartesian Plot

A plot of all the units in the game including those out of play is provided on each terminal's output. The format is the same as that for the players. The copies from both terminals should be identical. Two are provided for the convenience of the Control Group.

5. Data Base Printout

File FT02F001 is printed out on each terminal as it appears following the execution of TURN but without the updating accomplished in the CHANGE program. Each terminal's copy should be identical. Both copies are provided for Control Group use.

E. PROGRAM FAILURES

Only two kinds of program failures should occur in CASTAG. Either a data entry error in the interactive phases or a CP/CMS shutdown may cause termination of the program. The critical consideration in restarting CASTAG is that both programs in two different terminals must be kept executing together. Suggestions for dealing with some of the various possibilities are provided in this section.

1. Data Entry Error

Data entries to the EXEC routine cannot normally cause termination as illegal entries cycle the program back to the input statement. Entry of the wrong side, i.e., "R" vice "B" or wrong user ID will prevent any file transfers. To recover, terminate the EXEC and manually make the required transfer. Next turn the programs can be restarted together.

Although only three entries are required for SEARCH, these may have the most damaging effect as some illegal character entries are read as zero. If the TURN program terminates, the CASTAG EXEC may be restarted. "1" and "2" are the only entries accepted for Red/Blue terminal designation. Should both terminals enter the same number, stop execution and restart execution with "CASTAG EXEC". A time of zero will cause no movement or detections and again, the program may be restarted. The worst case is if the programs start with two different times. The original data base will be irrevocably changed to reflect movement. Short

of retyping the data base the only solution is to have the terminal with the shorter time execute SEARCH for the difference. The only effect on the game would be one more HFDF segment for the terminal executing twice.

CHANGE is where most errors usually occur due to the large number of entries required. All entries to CHANGE with the exception of the first may be changed again and again until the user terminates the program. The first entry which tells the program if it is Red or Blue will cause an error if the number does not match that entered in SEARCH. If CHANGE does terminate, it can be restarted with "CHANGE". The rest of the EXEC will have to be done manually until it is time to start the next turn. Then CASTAG EXEC can be restarted to get both terminals back together.

2. CP/CMS Failure

When a CP/CMS failure occurs, immediately check to see what condition file FT02F001 (the data base) is in. If both terminals were exactly in step and CP/CMS went down just after the file was rewound on both terminals but before the information was written back on the file, the data base will be lost.

Assuming the data base is intact on one terminal, execute CASTAG from the top utilizing the data base transfer section. The critical consideration is that both terminals always start from identical data bases.

V. FUTURE DEVELOPMENT AND RESULTS

A. IMPROVEMENTS AND EXTENSIONS OF CASTAG

Future work on CASTAG falls into three major categories. First, there is the inclusion of those parts of SEATAG not programmed. Second is improvements to facilitate play and umpire functions of the part of the game now covered. Last, improvements to the program to use less core and improve programming efficiency should be considered.

1. Tactical Improvements

In SEATAG, surface units with two air search radars are given no more capability than a unit with only one. Units with the dual capability should have some kind of enhanced detection capability against air targets.

The time constraint forced the omission of sonobuoy fields for ASW aircraft. These are treated as static detection areas in the game with a different detection methodology. Currently, they could be played manually in conjunction with CASTAG by the Control Group.

To make CASTAG a complete game requires the programming of the SEATAG engagement and damage methodology. In the author's opinion, an effort approximating the work done on this project would be required to accomplish this.

2. Display and Bookkeeping Improvements

The Control Group is required to monitor aircraft and diesel submarine time on station to prevent unrealistic

mission lengths. Time on station or time submerged could easily be kept in one of the unused columns of the DATA array.

To facilitate play of aircraft, the SEATAG aircraft carrier packages would speed play and reduce the number of units devoted to air assets. If the Blue player wants to use the SEATAG CV around the clock defensive package with a strike or cap surge, he will have to operate fifteen plus aircraft. The CV begins to approach the program limit of total units in a carrier strike warfare environment. In conjunction with programming the packages, multiple units where 10 strike aircraft would be one unit of composition 10 would be introduced.

The 1000 by 1000 mile plots should be made variable so the players can enter a midpoint and radius to designate the plot limits. Since most engagements take place at ranges less than 100 miles, a smaller scale output centered around the area of interest would be an improvement.

The player output should include a current range calculation whenever the (X,Y) position is given. As it now stands, the players have to make this determination themselves.

3. Programming Improvements

The input screening and interactive sequences have ample room for improvement. Some could be eliminated, particularly two of the three inputs telling the terminal whether it is Red or Blue.

The arrays RADAR, SONAR, ESMDDET, INTELL and HFDF are redundant. Their function could have been handled by one array.

There are, no doubt, other inefficiencies the discerning programmer could spot.

B. RESULTS

The SEATAG Documentation suggests lunch breaks and related lectures as a way to utilize the time the two sides will spend waiting for the umpires to finish. This is an example of the principal difficulty with a manual wargame, the time factor.

CASTAG reduces the time to play a turn of SEATAG to less than fifteen minutes. All of the time saved comes from what was essentially idle time for the players while the Control Group plotted the moves. The game can be used as is in the various wargaming courses extant at the Naval Postgraduate School and perhaps elsewhere.

APPENDIX A

ADDITIONAL UNITS

Ship Class /Aircraft Name: KRESTA I (4)						RED/BLUE/ Title: CLG						
CHARACTERISTICS	SENSOR	FUNCTION	RANGE *(nm)		SENSOR	FUNCTION	RANGE *(nm)					
			Good	Bad			Good	Bad				
FLD/MTOW(T): 8000	DON	SURF RADAR	20	15	PEEL GRP	(2) MFCS	-	-				
L(ft): 510	HEAD NET	C AIR RDR	150	75	MUFF COB	(2) GFCS	-	-				
W/B(ft): 56	BIG NET	AIR RADAR	150	75	BASS TILT	(2) GFCS	-	-				
D/H(ft): 20	PLINTH NET	?			SCOOP/	(1) SSM/MFCS	-	-				
Ceiling(kft):					PAIR							
Spd: Max(kt): 35	SONAR	ASW	10	2								
Cruise(kt):	WEAPONS	FUNCTION	# LAUNCHERS or MOUNTS	# BARRELS, RAILS PER SYS	TOTAL WPNS	RANGE *(nm)						
Patrol(kt): 18						Min	Max					
Mission						SS-N-3	ASUW/SSM	2	2	4	2	30#
Time(hr): 20 day						SA-N-1	AAW/SAM	2	2	60	2	16
Endurance						57mm GUN	AAW	2	2	1000	0	3
at Cruise(nm) 5500	20mm GUN	ASMD	4	-	2000	0	1					
Remarks:	MBU2500	ASW		12	48	0	3					
4 BOILERS	MBU4500	ASW	2	6	24	0	1					
TURBINES-2	21"TT	ASW/ASUW	2	5	10	2	10					
100,000 SHP		#170 NM	WITH EXTERNAL TARGETING									
	1	HORMONE A/B	HELICOPTERS									
NOTES: * These notational values are not necessarily precise. They are considered representative and provided for game purposes only.						"COST" 50 TOKENS						

Ship Class /Aircraft Name: KYNDA CLASS (4)						RED/BLUE Title: CLG		
CHARACTERISTICS	SENSOR	FUNCTION	RANGE *(nm)		SENSOR	FUNCTION	RANGE *(nm)	
			Good	Bad			Good	Bad
FLD/MTOW(T): 5700	DON	SURF RADAR	20	15	SCOOP/	(2) SSM/	-	-
L(ft): 468	HEAD NET	A AIR RDR	150	75	PAIR	MFCS	-	-
W/B(ft): 52	SONAR	ASW	10	2	PEEL/	(1) SAM/	-	-
D/H(ft): 17					GROUP	MFCS	-	-
Ceiling(kft):					OWL/	(1) 76mm	-	-
Spd: Max(kt):					SCREECH	GFCS		
Cruise(kt):	WEAPONS	FUNCTION	# LAUNCHERS or MOUNTS	# BARRELS, RAILS PER SYS	TOTAL WPNS	RANGE *(nm)		
Patrol(kt):						Min	Max	
Mission	SS-N-3	ASUW/SSM	2	4	16	2	30#	
Time(hr): 15 day	SA-N-1	AAW/SAM	1	2	30	2	16	
Endurance	76mm GUN	AAW	2	2	1000	0	5	
at Cruise(nm) 7000	MBU2500	ASW	1	12	24	0	3	
Remarks:	21" TT	ASW/ASUW	2	3	12	2	10	
4 BOILERS								
2 TURBINES		#170 NM WITH EXTERNAL TARGETING						
100,000 SHP								
NOTES: * These notional values are not necessarily precise. They are considered representative and provided for game purposes only.						"COST" 40 TOKENS		

APPENDIX B

CHANGE OUTPUT

EXECUTION BEINGS...

ENTER 1 FOR RED, 2 FOR BLUE, FORMAT 111.

>1

IT IS 360. Z TIME, YOU HAVE 6 UNITS.

ENTER THE ROW NUMBER OF THE UNIT YOU WISH TO CHANGE THE
STAUTS OF, 112, "0" WILL INDICATE NO CHANGES TO ANY UNIT.

>03

CODE UNIT ID= 52
2 SPEED IS 10 KIS.
3 ALT/DEPTH= 0*10**3 / 1=SHALLOW,2=DEEP.
4 COURSE is 80 T.
5 SURFACE SEARCH RADAR: 0 1=ON.
6 AIR SEARCH RADAR: 0 0=OFF.
7 SONAR: 0
8 HF COMMUNICATIONS: 0

ENTER THE CODE NUMBER OF THE LINE YOU WANT TO CHANGE,
111. " 0" MEANS NO CHANGES TO THIS UNIT.

>5

ENTER THE NEW VALUE FOR CODE NO. 5, 114.

>0001

CODE UNIT ID= 52
2 SPEED IS 10 KTS.
3 ALT/DEPTH= 0*10**3 / 1=SHALLOW,2=DEEP.
4 COURSE IS 80 T.
5 SURFACE SEARCH RADAR: 1 1=ON.
6 AIR SEARCH RADAR: 0 0=OFF.
7 SONAR: 0
8 HF COMMUNICATIONS: 0

ENTER THE CODE NUMBER OF THE LINE YOU WANT TO CHANGE,
111. " 0" MEANS NO CHANGES TO THIS UNIT.

>0

IT IS 360. Z TIME, YOU HAVE 6 UNITS.

ENTER THE ROW NUMBER OF THE UNIT YOU WISH TO CHANGE THE
STATUS OF, 112, "0" WILL INDICATE NO CHANGES TO ANY UNIT.

>0

IF YOU WISH TO BRING AN OUT OF PLAY UNIT
INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT),
ENTER THE ROW NUMBER OF THAT UNIT, 112. "0" MEANS
NO UNITS TO PUT IN OR OUT OF PLAY.

>05


```

CODE    UNIT ID=    0
  2     SPEED IS    0 KTS.
  3     ALT/DEPTH=  20*10**3 / 1=SHALLOW,2=DEEP.
  4     COURSE IS   90 T.

```

ENTER THE NEW UNIT ID ,114. "0" PUTS THE UNIT
OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO
PLAY WITH THAT ID NUMBER.

>0077

```

UNIT  5 IS NOW AT    900    500
ENTER FORMAT I4 FIRST THE NEW X COORDIANTE
THEN, AFTER THE SECOND APPEARS, THE NEW Y COORDINATE.

```

> 100

> 100

IF YOU WISH TO BRING AN OUT OF PLAY UNIT
INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT),
ENTER THE ROW NUMBER OF THAT UNIT, 112. "0" MEANS
NO UNITS TO PUT IN OR OUT OF PLAY.

> 3

```

CODE    UNIT ID=   52
  2     SPEED IS   10 KTS.
  3     ALT/DEPTH=  0*10**3 / 1=SHALLOW,2=DDEP.
  4     COURSE IS   80 T.

```

ENTER THE NEW UNIT ID ,114. "0" PUTS THE UNIT
OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO
PLAY WITH THAT ID NUMBER.

>0

IF YOU WISH TO BRING AN OUT OF PLAY UNIT
INTO PLAY OR PUT A UNIT OUT OF PLAY (I.E. SINK IT),
ENTER THE ROW NUMBER OF THAT UNIT, 112, "0" MEANS
NO UNITS TO PUT IN OR OUT OF PLAY.

>0

R; T=0.92/2.63 10.38.02

>

CAS TAG EXECUTIVE PROGRAM

```

&TYPEOUT OFF
&GLOBAL T
&PRINT THIS IS THE SEATAG EXEC ROUTINE.
&PRINT THIS PROGRAM WILL REQUIRE A DATA BASE
&PRINT WHICH HOLDS ALL THE APPROPRIATE SHIP AND
&PRINT AIRCRAFT CHARACTERISTICS.
-MSG2 &PRINT ENTER THE USER NUMBER OF THE OTHER PLAYER INCLUD-
&PRINT ING P OR G FOR USER TYPE, MAKING A TOTAL CF 5
&READ ARGS
&ID = &1
&ERFCR &CONTINUE
&PRINT IS OTHER PLAYERS ID = &ID ? ENTER Y OR N.
&READ ARGS
&IF &1 NE Y &GOTO -MSG3
&PRINT SENT TO THE CONSOLE OF THE OTHER PLAYER IF NC COPY
&PRINT EXISTS IN HIS FILES. IF YOU ARE THE OTHER PLAYER,
&PRINT YOU WILL HAVE TO READ FILE FTC2F001.
-MSG4 &PRINT IF NC TRANSFER OF DATA BASE REQUIRED ENTER N.
&PRINT TO TRANSFER DATA BASE TO OTHER PLAYER ENTER "T".
&PRINT TO READ THE DATA BASE FROM THE OTHER PLAYER ENTER "R".
&READ ARGS
&IF &1 EQ T &GOTO -TRANS
&IF &1 EQ N &GOTO -MSG5
&IF &1 NE R &GOTO -MSG4
-FILERD Y
CP QUERY F
&PRINT IF YOU HAVE A FILE SPOOLED WAITING TO BE READ, N.
&PRINT ENTER "Y". IF NOT ENTER ANY OTHER LETTER EXCEPT N.
&PRINT THE PROGRAM WILL CONTINUE TO RETURN HERE UNTIL THE
&PRINT FILE IS AVAILABLE OR UNTIL Y CR N IS ENTERED. TO
&PRINT GO AHEAD WITHOUT READING THE FILE ENTER N.
&READ ARGS
&IF &1 EQ N &GOTO -MSG5
&IF &1 NE Y &GOTO -FILERD
CP READ *
&GOTO -MSG5
-TRANS &CONTINUE
CP XFERC TO &ID
CP PUNCHCC FILE FTC2F001
CP XFERC OFF
-MSG5 &CONTINUE

```



```

CLOSED PRINTER OFF
FILEDEF 06 PTR
LOAD SEARCH (CLEAR XEQ NOMAP)
C CLOSE PRINTER ON
FILEDEF 06 CON
LOAD CHANGE (CLEAR XEQ NOMAP)
SUPRINT NOW EACH PLAYER SENDS HIS UPDATED FILE TO THE OTHER
-MSG6 &PRINT ENTER "R" IF YOU ARE RED, "B" IF BLUE.
GREAL ARGS
&IF &1 EQ R &SIDE = &1
&IF &1 EQ B &SIDE = &1
&PRINT YOU ENTERED &1 . IS THIS CORRECT? ENTER "Y" OR "N".
&IF &1 EQ N &GOTO -MSG6
&IF &SIDE EQ R &GOTO -RT0B
&IF &SIDE EQ B &GOTO -MSG6
-BT CR &CONTINUE
CP PUNCHCC FILE FT C4F001
CP XFER C OFF
&GOTO -MSG7
-RT CR &CONTINUE
CP XFER C TO &ID
CP PUNCHCC FILE FTC3F001
CP XFER D OFF
-MSG7 &CONTINUE
CP QUERY
&PRINT DO YOU HAVE A FILE SPOOLED TO BE READ? ENTER "Y" OR "N".
GREAL ARGS
&IF &1 EQ Y &GOTO -MSG7
&IF &1 EQ N *
LIST FILE *
&PRINT YOU SHOULD HAVE FILE TYPES FT02F001, FT03F001 , FTC4F001
&GOTO -MSG7
&PRINT YOU AND FT01F001 ON YOUR DISK AT THIS TIME.
LOAD UPDATE (CLEAR XEQ NOMAP)
ERASE FILE FTC3F001
ERASE FILE FT04F001
&PRINT TO PLAY ANOTHER TURN, ENTER Y. TO QUIT ENTER N.
GREAL ARGS
&IF &1 EQ Y &GOTO -MSG5
&EXIT

```


FORTRAN PROGRAM SEARCH

```

C
      INTEGER RADAR, SONAR, RDISK, BDISK, FPDF, DATA, ESMDET
      DIMENSION PRMAX(50,50), DICE(50), NROW(2), CPA(50,2)
      C PRADAR(50,20), SONAR(50,20), DATA(50,20), XPLOT(50), YPLOT(50),
      C FPDF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
      COMMON PRMAX, DATA, RADAR, SCNAR, HFCF, INTEL, ARCW, CPA,
      C NRED,NBLUE,NCOL,ID,ZTIME,EPS,ISEED1,ISEED2,ISEED3,
      C ICICE,NWRITE,NMAX,ESMDET,IFOE,CTIME,MINDEX,R,XPLCT,YFLCT
      SET INITIAL VALUES OF VARIABLES
      NSHIP = 50
      NTERM = 5
      NCCL = 15
      NMAX = 20
      ISEED1 = 1234567
      ISEED2 = 7654321
      ISEED3 = 9182736
      ICISK = 1
      BCISK = 3
      NRWITE = 4
      EPS = 5
      NREAD = COL
      NDISK = 2
      ENTER RN SEED, TIME OF TURN & WHETHER RED OR BLUE
      WRITE(NTERM,102)
      FORMAT('ENTER THE RANDOM NUMBER SEED, FORMAT 17.,/')
      102 READ(NREAD,103) KSEED
      FCRMAT(1,17) ISEED1+KSEED
      ISEED1 = ISEED2 + KSEED
      ISEED2 = ISEED3 + KSEED
      ISEED3 = ISEED4 + KSEED
      WRITE(NTERM,104)
      FCRMAT(1,ENTER) TIME
      104 READ(NREAD,105) TIME
      READ(NREAD,F4,C.)
      105 FCRMAT(F4,C)
      CTIME = TIME
      READ(NDISK,106) ZTIME,NRED,NBLUE
      106 FCRMAT(F6,C,2I3)
      NRCW(1) = NFED
      NROW(2) = NBLUE
      4 WRITE(NTERM,107) "1" IF YOU ARE THE RED PLAYER, "2" IF BLUE.
      107 FCRMAT(1,ENTER) ID
      C,, , FORMAT 111.,/)
      108 READ(NREAD,108) ID
      FCRMAT(111,

```



```

C
IF (IC -NE. 1 -ANC. ID -NE. 2) GC TO 4
IF (ID -EQ. 1) IFCE = 2
IF (ID -EQ. 2) IFCE = 1
ZERO OUT ARRAYS
DC 20 I=1, NSHIP
CPA (I,1) = 0.0
CPA (I,2) = 0.0
LICE(I) = C.0
CC 13 J=1, NCOL
RADAR(I,J) = 0
SONAR(I,J) = 0
DATA(I,J,1) = 0
DATA (I,J,2) = 0
CONTINUE
DO 14 K = 1, 4
MINDEX(K + 6) = 0
HFDF(I,K) = 0
CONTINUE
DO 15 L = 1, 6
INTEL(I,L) = 0
MINDEX(L) = 0
CONTINUE
DO 16 L = 1, 5
ESMDET (I,L) = 0
CONTINUE
CCNTINUE
READ DATA FROM DISK 2 INTO DATA ARRAYS
DC 30 I=1, NRED
READ (NDISK,101) (DATA (I,J,1), J=1, NCOL)
FCRMAT (15I4)
CCNTINUE
I=1, NBLUE
READ (NDISK,101) (DATA(I,J,2), J=1, NCOL)
CCNTINUE
IF THIS IS TURN 1, CALL NUMBER. IF NOT REAT ICISK
IF (ZTIME -LE. EPS. OR. TIME -LE. EPS) CALL NUMBER
IF (ZTIME -LE. EPS. OR. TIME -LE. EPS) GO TC 33
READ (IDISK,109) ((PRMAX(I,J,K), I = 1, NMAX), J= 1, NMAX), K = 1, 5)
FCRMAT (20F4.3)
REWIND IDISK IF OVER 24 HRS.
ZTIME = ZTIME + TIME
IF (ZTIME -GT. 1440.) ZTIME = ZTIME - 1440.
IF (ZTIME -GT. EPS) CALL MOVE(TIME) FILE AND CREATE FTC3 OR
WRITE UPDATED INFC BACK ON MASTER FTOL ALSO.
FT04 RESPECTIVELY. WRITE INFO CN IT FIRST.
REWIND DISK WITH DATA BASE CN IT FIRST.
WRITE(IDISK, 109) ((PRMAX(I,J,K), I=1, NMAX), J=1, NMAX), K= 1, 5)

```

```

DET00450
DET00460
DET00470
DET00480
DET00490
DET00500
DET00510
DET00520
DET00530
DET00540
DET00550
DET00560
DET00570
DET00580
DET00590
DET00600
DET00610
DET00620
DET00630
DET00640
DET00650
DET00660
DET00670
DET00680
DET00690
DET00700
DET00710
DET00720
DET00730
DET00740
DET00750
DET00760
DET00770
DET00780
DET00790
DET00800
DET00810
DET00820
DET00830
DET00840
DET00850
DET00860
DET00870
DET00880
DET00890
DET00900
DET00910
DET00920

```



```

REWINC NDISK
WRITE (NCISK,106) ZTIME,NRED,NBLUE
DC 35 I = 1,NRED
WRITE (NDISK,101) (DATA(I,J,1), J=1,NCOL)
35 CCNTINUE
DC 37 I = 1,NBLUE
WRITE (NDISK,101) (DATA(I,J,2), J=1,NCOL)
37 CCNTINUE
IF (ID.EQ. 2) GO TO 40
WRITE (RCISK,110) ZTIME,NRED
11C FCRMAT (F6.C,13)
DC 40 I = 1,NRED
WRITE (RDISK,101) (DATA(I,J,1), J = 1,NCCL)
40 CCNTINUE
IF (ID.EQ. 1) GO TO 97
WRITE (BDISK,110) ZTIME,NBLUE
DC 97 I = 1,NBLUE
WRITE (BDISK,101) (DATA(I,J,2), J=1,NCOL)
97 CCNTINUE
IF (TIME.GT. EPS) CALL DETECT (TIME)
R(1) = 1000.
R(2) = 0.0
R(3) = 1000.0
R(4) = 0.0
KTIME = ZTIME/60.
IMIN = 60.*(ZTIME/60.) - FLOAT(KTIME)
M = MINDEX(10)
NUNIT = NRQW(ID)
WRITE(6,55)
FCRMAT (I11)
55 NGRAPH = 3
IF (M.EQ. 0) NGRAPH = 1
IF (M.EQ. 0) GO TC 56
CALL UTPLT(XPLOT,YPLOT,M,R,1,1)
56 DC 57 I = 1,NUNIT
XPLOT(I) = DATA(I,5,ID)
YPLOT(I) = DATA(I,6,ID)
57 CCNTINUE
CALL UTPLT(XPLOT,YPLOT,NUNIT,R,1,NGRAPH)
WRITE(6,58) KTIME,IMIN
FCRMAT(I11, STATUS CF FORCES AS CF ,I3, MAX SPS, MIN,/,/,
C. UNIT SPEED DEPTH COURSE X Y TYPE MAX A/S SNR,/,/,
C. ID /HT DEG T GRID GRID
DC 61 I = 1,NUNIT
WRITE (6,6C) (DATA(I,J,ID),J=1,11)
6C FCRMAT ( , ,I4,2I5,3X,8I5)
61 CCNTINUE
WRITE (6,62)

```

```

DET0053C
DET00540
DET0055C
DET00560
DET00570
DET0058C
DET00590
DET01010
DET01020
DET01030
DET0104C
DET01050
DET0106C
DET01070
DET0108C
DET0109C
DET0110C
DET0111C
DET01120
DET01130
DET01140
DET0115C
DET0116C
DET0117C
DET01180
DET01190
DET0120C
DET01210
DET0122C
DET01230
DET0124C
DET0125C
DET01260
DET01270
DET01280
DET01290
DET01300
DET0131C
DET01320
DET0133C
DET01340
DET0135C
DET01360
DET01370
DET01380
DET01390
DET01400

```



```

62 FCRMAT(1H1)
CC 63 I=1, NRED
XPLCT(I) = DATA(I,5,1)
YPLCT(I) = DATA(I,6,1)
63 CCNTINUE
CALL UTPLCT(XPLCT,YPLCT,NRED,R,1,1)
CC 65 I=1,NBLUE
XPLCT(I) = DATA(I,5,2)
YPLCT(I) = DATA(I,6,2)
65 CCNTINUE
CALL UTPLCT(XPLCT,YPLCT,NBLUE,R,1,3)
999 STOP
ENC
SUBROUTINE MOVE(TIME)
INTEGER RADAR, SONAR, RDISK, BDISK, HPDF, DATA, ESMDET
DIMENSION PRMAX(50,5C,5), DICE(5C), NROW(2), CPA(5C,2),
C RADAR(50,2C), SONAR(50,20), DATA(50,20,2), XPLCT(50), YPLCT(50),
C HPDF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
C COMMON PRMAX, DATA, RADAR, SONAR, HPDF, INTEL, NROW, CPA,
C NRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEED1, ISEED2, ISEED3,
C DICE, NWRITE, NMAX, ESMDET, IFOE, CTIME, MINDEX, R, XPLCT, YPLCT
TFOUR = TIME/60.0
RAD 10 K=1,2
DC 5 I=1, NBLUE
CIST = FLOAT(DATA(I,2,K))*THCUR
THETA = FLOAT(DATA(I,4,K))*RAD
DATA(I,5,K) = DATA(I,5,K) + (.5 *DIST*SIN(THETA))
DATA(I,6,K) = DATA(I,6,K) + (.5 *DIST*COS(THETA))
IF (DATA(I,5,K) .LT. 0) DATA(I,5,K) = 0
IF (DATA(I,6,K) .LT. 0) DATA(I,6,K) = 0
CCNTINUE
10 CCNTINUE
CC RETURN
ENC
SUBROUTINE NUMBER
INTEGER RADAR, SONAR, RDISK, BDISK, HPDF, DATA, ESMDET
DIMENSION PRMAX(50,5C,5), DICE(5C), NROW(2), CPA(50,2),
C RADAR(50,2C), SONAR(50,20), DATA(50,20,2), XPLCT(50), YPLCT(50),
C HPDF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
C COMMON PRMAX, DATA, RADAR, SONAR, HPDF, INTEL, NROW, CPA,
C NRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEED1, ISEED2, ISEED3,
C DICE, NWRITE, NMAX, ESMDET, IFOE, CTIME, MINDEX, R, XPLCT, YPLCT
DC 30 K=1,5
CO 20 J=1, NMAX
CALL RANDOM(ISEED1, DICE, 50)
CO 10 I=1, NMAX
PRMAX(I,J,K) = 0.0

```

DET01410
 DET01420
 DET01430
 DET01440
 DET01450
 DET01460
 DET01470
 DET01480
 DET01490
 DET01500
 DET01510
 DET01520
 DET01530
 DET01540
 DET01550
 DET01560
 DET01570
 DET01580
 DET01590
 DET01600
 DET01610
 DET01620
 DET01630
 DET01640
 DET01650
 DET01660
 DET01670
 DET01680
 DET01690
 DET01700
 DET01710
 DET01720
 DET01730
 DET01740
 DET01750
 DET01760
 DET01770
 DET01780
 DET01790
 DET01800
 DET01810
 DET01820
 DET01830
 DET01840
 DET01850
 DET01860
 DET01870
 DET01880


```

IF (DICE(I) -GE. .9) GO TO 1C
PRMAX (I,J,K) = .9 * SQR(1.0 - (CICE(I)**2/.81))
CONTINUE
CONTINUE
CONTINUE
RETURN
END
SUBROUTINE DETECT(TIME)
INTEGER RADAR, SONAR, ROISK, BDISK, FDF, DATA, ESMDET
DIMENSION PRMAX (50,50), DICE(50), NROW(2), CPA(50,2), XPLOT(50), YPLOT(50),
C RADAR(50,20), SONAR(50,20), DATA(50,20,2), MINDEX(10), R(4)
C FDF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
C CMCN PRMAX, NCOT, RADAR, SCNAR, FCLF, INTEL, ARCW, CPA, ISEED1, ISEED2, YFLCT
C NBLUE, NWRITE, NMAX, EPS, ISEED1, ISEED2, YFLCT
C DICE, IF THIS TURN INCLUDES AN INTELL/SURV SEGMENT
C DETERMINE ZTIME - TIME
Z2 = ZTIME
DC 11 NX = 14
PR = 360.0*(NX-1)
IF (ZTIME -GE. HR .AND. Z2 .LE. PR) GO TO 12
CONTINUE
11 GO TO 13
CALL SPOOK
12 DETERMINE ANY INTELLIGENCE FROM HF COMMS
13 CALL RANDOM (ISEED2, DICE, 50)
NN = 0
NN = NRCW(IFOE)
DC 16 I = 1,NN
IF (DATA(I,1,IFOE) .LE. 0) GO TO 16
IF (DATA(I,15,IFOE) .LE. 0) GO TO 16
IF (DICE(I) .LE. .79) GO TO 16
N = N+1
FDF(N,1) = DATA(I,1,IFOE)
FDF(N,2) = DATA(I,7,IFOE)
FDF(N,3) = DATA(I,5,IFOE)
FDF(N,4) = DATA(I,6,IFOE)
XPLOT(N) = DATA(I,5,IFOE)
YPLOT(N) = DATA(I,6,IFOE)
CONTINUE
16 MINDEX(10) = N
IF (N .EQ. C) GO TO 23
WRITE (NWRITE,18)
FCRMT (, ,15X, 'HF DETECTIONS OF ENEMY CCMS',/,14X,
C , X COORD ',/)
DC 20 I = 1,N
WRITE(NWRITE,19) (HDF(I,J),J=3,4)
FCRMT (, ,12X,216,/)
CONTINUE
15
16

```

DET0189C
 DET0190C
 DET0191C
 DET0192C
 DET0193C
 DET0194C
 DET0195C
 DET0196C
 DET0197C
 DET0198C
 DET0199C
 DET0200C
 DET0201C
 DET0202C
 DET0203C
 DET0204C
 DET0205C
 DET0206C
 DET0207C
 DET0208C
 DET0209C
 DET0210C
 DET0211C
 DET0212C
 DET0213C
 DET0214C
 DET0215C
 DET0216C
 DET0217C
 DET0218C
 DET0219C
 DET0220C
 DET0221C
 DET0222C
 DET0223C
 DET0224C
 DET0225C
 DET0226C
 DET0227C
 DET0228C
 DET0229C
 DET0230C
 DET0231C
 DET0232C
 DET0233C
 DET0234C
 DET0235C
 DET0236C


```

C      CHECK PLATFORM VS PLATFORM DETECTIONS
23     LIGHT = 0
        IF (ZTIME .GT. 360. .AND. ZTIME .LT. 1080.) LIGHT = 1
        IUNIT = NROW(ID)
        DO SC I = 1, IUNIT
            IF (DATA(I,1,1,1)) .EQ. 0) GO TO 5C
            ITYPE = DATA(I,7,1,1) -- (1000 * IC)
            IS UNIT A SUBMARINE?
            IF (ITYPE .GE. 200) .OR. ITYPE .LT. 100) GO TC 40
            CALL NLPCPA(I, TIME)
            IF SONAR PASSIVE AND SPEED LE 22 KNOTS CALL PASSIVE
            IF (DATA(I,2,1,1)) .GE. 23) GO TC 25
            IF (DATA(I,14,1,1)) .EQ. 0) CALL PSONAR(I,1)
            IS SUB DEEP?
            IF (DATA(I,3,1,1)) .EQ. 2) GO TO 30
            CALL ESM(I)
            IF SURFACE SEARCH RADAR IS ON & SLE NOT DEEP...
            IF (DATA(I,12,1,1)) .EQ. 1) CALL DETRAD(I)
            IS SUB SURFACED & IT IS DAY CHECK VISUAL SIGHTINGS
            IF (DATA(I,3,1,1)) .EQ. 0 .AND. LIGHT .EQ. 1) CALL VISUAL(I, ITYPE)
            IF SONAR ACTIVE AND SPEED LE 15 KTS
            IF (DATA(I,14,1,1)) .EQ. 1 .AND. DATA(I,2,1,1) .LE. 15) CALL ASONAR(I)
            GO TO 50
            IS UNIT A SURFACE SHIP?
            IF (ITYPE .LT. 300) GO TO 50
            CALL NLPCPA(I, TIME)
            CALL ESM(I)
            IF (LIGHT .EQ. 1) CALL VISUAL(I, ITYPE)
            IS EITHER AIR OR SURFACE SEARCH RADAR ON?
            IF (DATA(I,12,1,1)) .EQ. 1 .OR. DATA(I,13,1,1) .EQ. 1) CALL DETRAD(I)
            IF SONAR PASSIVE AND SPEED LESS THAN 17) CALL PASSIVE SONAR
            IF (DATA(I,14,1,1)) .EQ. 0 .AND. DATA(I,2,1,1) .LE. 17) CALL PSONAR(I,2)
            IF SONAR ACTIVE AND SPEED LT 15 KNOTS, CALL ACTIVE SONAR
            IF (DATA(I,14,1,1)) .EQ. 1 .AND. DATA(I,2,1,1) .LE. 15) CALL ASCNAR(I)
            GO TO 90
5C     ITYPE = ITYPE - 200
        IF IT IS DAY, ALL AIRCRAFT CALL VISUAL
        CALL NLPCPA(I, TIME)
        IF (LIGHT .EQ. 1) CALL VISUAL(I, ITYPE)
        IF AIRCRAFT IS FIGHTER/ ATTACK GO TC 90
        IF (ITYPE .LT. 70) GO TC 90
        ALL OTHER TYPES MAKE ESM & RADAR DETECTIONS
        CALL ESM(I)
        CALL DETRAD(I)
        ONLY SH-3 & HORMONE MAY USE DIPPING SCNAR IF SPEED LT 10 & ALT LT 1
        IF (ITYPE .EQ. 93 .AND. DATA(I,2,1,1) .LE. 10 .AND.
            DATA(I,3,1,1) .LE. 1) CALL ASCNAR(I)
5C     CCNTINUE

```

DET0237C
DET02380
DET0239C
DET0240C
DET02410
DET0242C
DET02430
DET02440
DET02450
DET02460
DET02470
DET02480
DET02490
DET02500
DET0251C
DET02520
DET0253C
DET02540
DET0255C
DET02560
DET02570
DET02580
DET02590
DET02600
DET02610
DET0262C
DET02630
DET02640
DET02650
DET0266C
DET0267C
DET02680
DET02690
DET02700
DET0271C
DET02720
DET02730
DET02740
DET02750
DET02760
DET02770
DET0278C
DET0279C
DET0280C
DET02810
DET0282C
DET02830
DET02840


```

SS RETURN
END
SUBROUTINE SPOOK
  INTEGER RADAR, SONAR, RDISK, BDISK, FPDF, DATA, ESMDET
  DIMENSION PRMAX(50,20), SCNAR(50,6), DATA(50,20,2), CPLOT(50), YPLOT(50),
  C PDEF(50,4), INTEL(50,6), ESMDET(100,5), MINDEX(10), R(4)
  C COMMON PRMAX, DATA, RADAR, SCNAR, HPCF, INTEL, ARCW, CPA,
  C NRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEED1, ISEED2, ISEED3, YFLOT
  C DICE, NWRITE, NMAX, ESMDET, IFOE, DTIME, MINDEX, R, XPLCT, YFLOT
  THIS SUBROUTINE DETERMINES THE 0001,0600, 12CC
  & 1800 INTELLIGENCE INFORMATION.
  NUNIT = NROW(IFOE)
  N = 0
  CALL RANDOM (ISEED3, DICE, NUNIT)
  CC 11 I = 1, NUNIT
  IF (DICE(I) .LT. .59) GO TO 11
  IF (DATA(I,1,IFOE) .LE. 0) GO TO 11
  MCCN = 0
  ID1 = DATA(I,7,IFOE) - (1000 * IFOE)
  CC 9 J = 12,15
  MCCN = MCCN + DATA(I,J,IFOE)
  CC 5 CONTINUE
  IF (DICE(I) .LT. .79 .AND. MCCN .LE. 0) GO TO 11
  IS UNIT SUB UNDER 15 KTS?
  IF (ID1 .LT. 200 .AND. DATA(I,2,IFOE) .LT. 15) GO TO 11
  IS UNIT AIRCRAFT IN EMCON?
  IF (ID1 .LT. 300 .AND. MCCN .LE. 0) GO TO 11
  AN INTELLIGENCE/SURVEILLANCE DETECTION HAS OCCURED
  N = N+1
  IF (DICE(I) .GT. .8) GO TO 10
  C ADDITIONAL INFC
  IC2 = FL CAT(DATA(I,7,IFOE)/100)
  INTEL(N,5) = DATA(I,7,IFOE) - ID2*100
  INTEL(N,6) = ((DATA(I,4,IFOE) + 5)/10.)*10
  C BASIC INFORMATION
  INTEL(N,1) = DATA(I,1,IFOE)
  INTEL(N,2) = FL CAT(DATA(I,7,IFOE)/100)
  INTEL(N,3) = DATA(I,5,IFOE)
  INTEL(N,4) = DATA(I,6,IFOE)
  XPLCT(N + MINDEX(10)) = DATA(I,5,IFOE)
  YPLOT(N + MINDEX(10)) = DATA(I,6,IFOE)
  CC 11 CONTINUE
  MINDEX(10) = MINDEX(10) + N
  IF (N .EQ. 0) GO TO 99
  C WRITE (NWRITE,12)
  C FCRMAT(10X,'UNIT TYPE',3X,'X CCRD YCOORD',, PLATFORM COURSE')
  C WRITE (NWRITE,13) ((INTEL(I,J), J=2,6), I=1,N)

```


DET0425C
DET04300
DETC4310
DET04320
DET04330
DET04340
DET04350
DET04360
DET04370
DET04380
DET04390
DET04400
DET04410
DET04420
DET04430
DET04440
DET04450
DET04460
DET04470
DET04480
DET04490
DET04500
DET04510
DET04520
DET04530
DET04540
DET04550
DET04560
DET04570
DET04580
DET04590
DET04600
DET04610
DET04620
DET04630
DETC4640
DET04650
DET04660
DET04670
DET04680
DET04690
DET04700
DET04710
DET04720
DET04730
DET04740
DETC4750
DET04760

```

RADAR(M,8) = 1
RADAR(M,9) = CPA(N,1)
RADAR(M,10) = CPA(N,2)
XPLOT(M + MINDEX(10)) = DATA(N,5,IFOE)
YPLOT(M + MINDEX(10)) = DATA(N,6,IFOE)
GO TO 50
IF AIR SEARCH RADAR CN AND TGT IS AN AIRCRAFT.
25 IF (DATA(I,13,10) .LE. 0) GO TO 50
IF (ITGT .LT. 200 .OR. ITGT .GE. 300) GO TC 50
ARANGE = PRMAX(I,N,2) * DATA(I,10,10)
IF (CPA(N,1) .GT. ARANGE) GO TC 50
N = M+1
CO 30 J = 1,7
RADAR(M,J) = DATA(N,J,IFOE)
30 CCNTINUE
XPLOT(M + MINDEX(10)) = DATA(N,5,IFOE)
YPLOT(M + MINDEX(10)) = DATA(N,6,IFOE)
RADAR(M,3) = 13
RADAR(M,2) = DATA(N,7,IFOE)
RADAR(M,7) = DATA(N,2,IFCE)
RADAR(M,8) = 1
RADAR(M,9) = CPA(N,1)
RADAR(M,10) = CPA(N,2)
50 CCNTINUE
MINDEX(10) = MINDEX(10) + M
IF (M .EQ. C) GO TO 99
PRINT OUT RESULTS FOR UNIT I
WRITE(NWRITE,60) DATA(I,1,10)
60 FCRMAT(/,/,RADAR DETECTIONS BY UNIT ',I4,/,
C8X, ' RADAR TYPE ',2X, ' COURSE ',3X, 'X COORD YCCRD ',2X, 'SPEED ',
C3X, 'COMP. ',/)
DC 75 II = 1,M
WRITE (NWRITE,70) (RADAR(II,JJ),JJ=3,8)
70 FCRMAT (/,/,8X,6I8)
75 CCNTINUE
WRITE (6,76) DATA(I,1,10)
76 FCRMAT (/,/, ' UMPIRE VERSION OF RADAR DETECTIONS, UNIT ',I4,/,
C4X, ' ID ',2X, ' TYPE ',RADAR, '3X, 'CSE ', 'X GRID Y GRID ',
C 'SPEED ',1X, 'COMP. ',CPA RGE & ERG ')
DC 78 II = 1,M
WRITE (6,77) (RADAR(II,JJ), JJ = 1,10)
77 FCRMAT (/,/,8I6,3X,2I6)
78 CCNTINUE
79 RETURN
99
END
SLROUTINE ASONAR(I)
INTEGER RADAR, SONAR, ROISK,8DISK,1FDF,DATA,ESMDET
DIMENSION PRMAX (50,50),DICE(50),NROW(2),CPA(50,2),

```



```

C RADAR(50,20), SONAR(50,20), DATA(50,20,2),XPLOT(50),YPLCT(50),
C FPDF(50,4), INTEL(50,6), ESMDET(100,5),MINDEX(10), R(4)
C COMMON PRMAX, DATA, RADAR, SCNAR, FFCF, INTEL, NRCW, CPA,
C NRED, NBLUE, NCOL, ID, ZTIME, EPS, ISEED1, ISEED2, ISEED3, YFLCT
C LICE, NWRITE, NMAX, ESMDET, IFOE, CTIME, MINDEX, R, XPLOT, YPLCT
C THIS SUBROUTINE CALCULATES ALL POSSIBLE DETECTIONS
FROM ACTIVE SONAR BY THE ITH UNIT. SONAR IS
CHECKED TO BE ACTIVE AND SPEED LESS THAN 15 KNOTS
PRIOR TO CALLING THE ROUTINE. BOTH RED AND BLUE SUBS MAY BE
DETECTED REGARDLESS OF WHETHER I IS RED OR BLUE. THE
DATA FROM NUMBER AND NLPCA MUST BE AVAILABLE. THIS
SUBROUTINE IS CALLED IN ARRAY SONAR.
DETECTIONS ARE STORED IN ARRAY SONAR.
NUNIT = NROW(IFOE)
ISAVE = IFOE
KUNIT = NROW(ID)
M = 0
DO 60 K = 1,2,NUNIT
DO 50 N = 1,NUNIT
IF (DATA(N,1,IFOE) .EQ. 0) GO TO 50
IF (I .EQ. N .AND. K .EQ. 2) GO TO 50
IS THE TARGET A SUBMARINE NOT ON THE SURFACE?
ITGT = DATA(N,7,IFOE) - (1000 * IFOE)
IF (DATA(N,3,IFOE) .EQ. 0 .OR. ITGT .GE. 200) GO TO 50
IF RANGE = PRMAX(I,N,3 + K) * DATA(I,11,ID)
IF (CPA(N,1) .GT. RANGE) GO TO 50
A DETECTION HAS OCCURED
M = M+1
GO 40
IA = 1,6
SONAR(M,IA) = DATA(N,IA,IFOE)
40 CONTINUE
SCNAR(M,2) = DATA(N,7,IFOE)
SCNAR(M,7) = DATA(N,2,IFOE)
SCNAR(M,8) = CPA(N,1)
SCNAR(M,9) = CPA(N,2)
50 CONTINUE
IF (K .EQ. 2) GO TO 60
NUNIT = KUNIT
IF N = ID
TIME = DTIME
CALL NLPCA(I, TIME)
60 CONTINUE
IFCE = ISAVE
IF (M .EQ. 0) GO TO 99
IF PRINT CUT SONAR
WRITE(NWRITE,70) DATA(I,1,ID)
70 FORMAT (/,/, ACTIVE SONAR DETECTIONS BY UNIT ,I4,/,17X
C , , DEPTH ,3X, COURSE ,4X, X COCRC ,1X, Y CCCRD ,2X, ,SPEED ,)

```

```

DETC477C
DETC478C
DETC479C
DETC480C
DETC481C
DETC482C
DETC483C
DETC484C
DETC485C
DETC486C
DETC487C
DETC488C
DETC489C
DETC490C
DETC491C
DETC492C
DETC493C
DETC494C
DETC495C
DETC496C
DETC497C
DETC498C
DETC499C
DETC500C
DETC501C
DETC502C
DETC503C
DETC504C
DETC505C
DETC506C
DETC507C
DETC508C
DETC509C
DETC510C
DETC511C
DETC512C
DETC513C
DETC514C
DETC515C
DETC516C
DETC517C
DETC518C
DETC519C
DETC520C
DETC521C
DETC522C
DETC523C
DETC524C

```



```

DC 85 II = 1,M
WRITE(NWRITE,80)(SONAR(II,JJ),JJ = 3,7)
FORMAT(1,14X,217,6X,3I7)
CCCONTINUE
80 WRITE(6,86) DATA(I,1,ID)
85 FCFMAT(1,1,1,ACTIVE,SONAR,ARRAY,UMPRIRE,VERSION,UNIT,I4,/,
C 2X,CPA,DI,ST, & BRC:)
C 2X,88 II = 1,M
DC 88 II = 1,M
WRITE(6,87)(SONAR(II,JJ), JJ = 1,9)
87 FCFMAT(1,1,7I6,2X,2I6)
XPLOT(MINDEX(10) + II) = SONAR(II,5)
YPLCT(MINDEX(10) + II) = SONAR(II,6)
88 CCCONTINUE
SS MINDEX(10) = MINDEX(10) + M
RETURN
END
SLROUTINE NLPCPA(I,TIME)
INTEGER RADAR,SONAR, RDISK, RDISK, FPDF, DATA, ESMDET
DIMENSION PRMAX(50,5), DICE(50),NR0W(2),CPA(50,2),YPLCT(50),
C FPDF(50,2),INTEL(50,20),DATA(50,20),XPLCT(50),R(4)
C COMMON PRMAX, DATA, ID, ZTIME, SONAR, FPDF, INTEL, NR0W, CPA, ISEED3, YPLCT
C LICE, NWRITE, NMAX, NCOL, ID, ZTIME, EPS, ISEED1, ISEED2, R, XPLCT, YPLCT
C FOR THE CPA UNIT DETERMINING THE RANGE AND BEARING
C POSING BEARING IN DEGREES TRUE FROM THE ITH UNIT.
ANC B = BEARING IN DEGREES TRUE FROM THE ITH UNIT.
RAC = 3.14159/ 180.0
T = TIME/60.
NUNIT = NR0W(IFOE)
A = FLOAT(DATA(I,5,ID))
B = FLOAT(DATA(I,6,ID))
D2 = T * RAD * FLOAT(DATA(I,2,ID))
ETA = RAD * D2 * SIN(ETA)
XCIST = D2 * COS(ETA)
YCIST = D2 * SIN(ETA)
DC 50 N = 1, NUNIT
CALCULATE TIME, X2, THE RELATIVE POSITION OF THE NTH UNIT
THETA = TIME, X2, THE RELATIVE POSITION OF THE NTH UNIT
X1 = DATA(N,4,IFOE)) * RAD
X2 = DATA(N,5,IFOE)
Y2 = DATA(N,6,IFOE)
X1 = X2 + XCIST - C1 * SIN(THETA)
Y1 = Y2 + YCIST - D1 * COS(THETA)
CHECK IF SLOPE OF DRN LINE IS INFINITE

```

DETO5250
DETO5260
DETO5270
DETO5280
DETO5290
DETO5300
DETO5310
DETO5320
DETO5330
DETO5340
DETO5350
DETO5360
DETO5370
DETO5380
DETO5390
DETO5400
DETO5410
DETO5420
DETO5430
DETO5440
DETO5450
DETO5460
DETO5470
DETO5480
DETO5490
DETO5500
DETO5510
DETO5520
DETO5530
DETO5540
DETO5550
DETO5560
DETO5570
DETO5580
DETO5590
DETO5600
DETO5610
DETO5620
DETO5630
DETO5640
DETO5650
DETO5660
DETO5670
DETO5680
DETO5690
DETO5700
DETO5710
DETO5720


```

C
IF (ABS(X2-X1) .LE. EPS) GO TO 16
CALCULATE SLCPE CF DRM
DX = (Y2 - Y1)/(X2-X1)
GC TO 18
16 IF (ABS(Y2-Y1) .LT. EPS) GO TO 3C
X = X2
Y = Y2
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 35
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 3C
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 3C
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 35
GC TO 40
18 X = (X + B*(X + X1)*DX*(X - DX*Y1)/(1.0 + DX **2))
Y = (Y - X1)*DX + Y1
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 35
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 3C
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 3C
IF (B .GT. Y1 .AND. B .GT. Y2 .AND. Y1 .GT. Y2) GC TC 35
GC TO 4C
3C X = X2
Y = Y2
GC TO 40
35 X = X1
Y = Y1
CALCULATE DISTANCE TO (X,Y) FROM (A,B)
CPA(N,1) = SQRT ((X-A)**2 + (Y-B)**2)
IF CPA IS VERY "SMALL", SET IT TO EPS
IF (CPA(N,1) .LT. EPS) CPA(N,1) = EPS
TAU = ARCSIN ((X-A)/CPA(N,1))
IF ((X .GT. A) .AND. (Y .GE. B)) TAU = TAU + 2.0 * 3.14159
IF ((X .GT. A) .AND. (Y .GE. B)) TAU = 3.14159 - TAU
IF ((X .EQ. A) .AND. (Y .LT. B)) TAU = 3.14159
IF ((X .LT. A) .AND. (Y .LT. B)) TAU = - TAU
CPA(N,2) = TAU/RAD
CCNTINUE
CC RETURN
END
SUBROUTINE VISUAL(I, ITYPE)
INTEGER RADAR, SONAR, RDISK, BDISK, FDOF, DATA, ESMDET
DIMENSION PRMAX(50,5), DICE(5C), NROW(2), CPA(50,2), YPLCT(50),
C FADAR(5C,2C), SONAR(50,20), DATA(50,20), XPLCT(50), R(4),
C FFCF(50,4), INTEL(50,6), ESMDET(100,5), MINCEX(10), R(4)
COMMON PRMAX, DATA, RADAR, SCNAR, HFCF, INTEL, NRCW, CPA,
C NRCM, NBLUE, NCOL, ID, ZTIME, EPS, ISEED1, ISEED3,
C DICE, NWRITE, NMAX, ESMDET, IFOE, CTIME, MINDEX, R, XPLCT, YPLCT
THIS SUBROUTINE CALLED FROM DETECT.CPA,S FRCN

```



```

52 B2 = 2.0
   C = 5
   A = FLOAT ( DATA(N,2,IFOE) )/5. + 15.
   GC TO 54
53 B2 = (FLOAT(DATA(N,2,IFOE))- 20.C) / (-2.0)
   IF (DATA(N,2,IFOE) .LE. 20) B2 = 0.0
   IF (DATA(I,2,ID) .GT. 18) GC TO 60
   C = 4
   IF (DATA(N,2,IFOE) .GT. 15) C = 6
   A = C * FLCAT(15 - DATA(N,2,IFOE)) + DATA(N,2,IFOE)
54 B = DATA(N,2,IFOE) + B2
   X = DATA(I,2,ID)
   IF (X .GT. A) A = X
   Y = SQRT(B*2*((1.0-((X/A)**2)))
   IF (Y .GT. DATA(I,11,ID)) Y = DATA(I,11,ID)
   RANGE = Y * PRMAX(I,N,5)
   IF (CPA(N,1) .GT. RANGE) GO TO 6C
   MM = MM+1
   CC 55 IK = 1,7
   SCNAR(MM,IK + 10) = DATA(N,IK,IFCE)
55 CCNTINUE
   SCNAR(MM,18) = CPA(N,1)
   SCNAR(MM,19) = CPA(N,2)
56 CCNTINUE
   IF (M .EQ. MM) GO TO 99
   WRITE(NWRITE,90) DATA(I,1,ID)
90 FCRMAT(/,/, , PASSIVE DETECTIONS BY UNIT',I4,/,4X,
   C , COURSE',2X, SPEED',3X, DEPTH',3X, RANGE',1X, BEARING')
   C M1 = M+1
   CC 93 II = M1,MM
   WRITE(NWRITE,92) SONAR(II,14),SCNAR(II,12) ,SCNAR(II,13),
   C SONAR(II,11),SONAR(II,19)
52 FCRMAT(/,/,3I8,3X,2I6)
93 CCNTINUE
   WRITE(6,94) DATA(I,1,ID)
54 FCRMAT(/,/, , UMPIRE COPY, PASSIVE SONAR DETECTIONS, UNIT ',I4,
   C /,4X, ID =M1,MM
   DC 96 II =M1,MM
   WRITE(6,95) (SONAR (II,JJ), JJ = 11,19)
55 FCRMAT ( , , ,2I5,2X,6I5)
56 CCNTINUE
55 RETURN

```

```

DETO7170
DETO7180
DETO7190
DETO7200
DETO7210
DETO7220
DETO7230
DETO7240
DETO7250
DETO7260
DETO7270
DETO7280
DETO7290
DETO7300
DETO7310
DETO7320
DETO7330
DETO7340
DETO7350
DETO7360
DETO7370
DETO7380
DETO7390
DETO7400
DETO7410
DETO7420
DETO7430
DETO7440
DETO7450
DETO7460
DETO7470
DETO7480
DETO7490
DETO7500
DETO7510
DETO7520
DETO7530
DETO7540
DETO7550
DETO7560
DETO7570
DETO7580
DETO7590
DETO7600
DETO7610
DETO7620
DETO7630

```


CHA0001C
CHA00020
CHA00030
CHA00040
CHA00050
CHA00060
CHA0007C
CHA00080
CHA00090
CHA00100
CHA00110
CHA00120
CHA00130
CHA00140
CHA00150
CHA00160
CHA00170
CHA00180
CHA00190
CHA00200
CHA00210
CHA00220
CHA00230
CHA00240
CHA00250
CHA00260
CHA0027C
CHA00280
CHA0029C
CHA00300
CHA00310
CHA00320
CHA00330
CHA00340
CHA00350
CHA0036C
CHA00370
CHA00380
CHA00390
CHA00400
CHA00410
CHA00420
CHA00430
CHA0044C

CHA0001C
CHA00020
CHA00030
CHA00040
CHA00050
CHA00060
CHA0007C
CHA00080
CHA00090
CHA00100
CHA00110
CHA00120
CHA00130
CHA00140
CHA00150
CHA00160
CHA00170
CHA00180
CHA00190
CHA00200
CHA00210
CHA00220
CHA00230
CHA00240
CHA00250
CHA00260
CHA0027C
CHA00280
CHA0029C
CHA00300
CHA00310
CHA00320
CHA00330
CHA00340
CHA00350
CHA0036C
CHA00370
CHA00380
CHA00390
CHA00400
CHA00410
CHA00420
CHA00430
CHA0044C


```

IF (NCODE .LE. 1 .OR. NCODE .GT. 8) GO TO 35
WRITE (NWRITE,50) NCODE VALUE FCR CODE NO.,12,' 114.',/)
50 FCRMAT (, ENTER THE NEW VALUE FCR CODE NO.,12,' 114.',/)
IF (NCODE .GT. 4) NCODE = NCODE+7
READ (NREAD,52) DATA (N,NCODE)
52 FCRMAT (114)
IF (DATA(N,NCODE) .LT. 0) DATA (N,NCODE) = C
IF (NCCDE .EQ.2) .ANC. (DATA (N,NCODE) .GT. DATA(N,8))
C DATA (N,NCCDE) = DATA(N,8)
IF (NCODE .EQ. 4 .AND. DATA(N,NCODE) .GE. 300) DATA (N,NCODE) = 0
ITYPE = DATA(N,7) -(ICENT * 1000)
IF (NCODE .EQ. 3 .AND. DATA(N,CCDE) .NE. C .AND. ITYPE .GE. 300)
C DATA (N,NCODE) = 0
CC TO 45
55 WRITE (NWRITE,60)
60 FCRMAT (, IF YOU WISH TO BRING AN CUT OF PLAY UNIT',/,/,
C , INTO PLAY OR PUT A UNIT OUT OF PLAY ( I.E. SINK IT),',/,
C , ENTER THE ROW NUMBER OF THAT UNIT, 112. "C" MEANS',/,
C , NO UNITS TO PUT IN OR OUT OF PLAY.',/)
READ (NREAD,42) IDNO
IF ( IDNO .EQ. 0) GO TO 90
IF ( IDNO .GT. NRCW) GO TC 59
WRITE (NWRITE,46) (DATA( ICNO,J),J =1,4)
WRITE (NWRITE,66)
66 FCRMAT (/, ENTER THE NEW UNIT ID,114. "C" PUTS THE UNIT',/,
C , OUT OF PLAY. ANY OTHER VALUE BRINGS THE UNIT INTO ',/,
C , PLAY WITH THAT ID NUMBER.',/)
READ (NREAD,68) DATA( IDNO,1)
68 FCRMAT (114)
IF (DATA( ICNO,1) .NE. 0) GO TO 7C
DATA( ICNO,5) = 0
DATA( IDNO,6) = 0
GC TO 59
7C WRITE (NWRITE,72) ICNO,DATA( ICNO,5), DATA( ICNO,6)
72 FCRMAT (, UNIT,13, IS NOW AT,215,/,
C , ENTER FORMAT 14, FIRST THE NEW X CO-ORDINATE ',/,
C , THEN, AFTER THE SECOND > APPEARS, THE NEW Y CO-ORDINATE.',/)
READ (NREAD,74) (DATA( IDNO,J),J=5,6)
74 FCRMAT (114)
GC TO 59
9C REWIND IDISK
WRITE (IDISK,30) ZTIME,NROW
DC 92 I=1,NROW
WRITE (IDISK,31) (DATA(I,J),J=1,NCOL)
52 STOP
55 END

```


FORTRAN PROGRAM UPDATE

```

1C      INTEGER  RDATA,BDATA,BDISK,RDISK
        DIMENSION RCATA(50,20), BDATA(50,2C)
        NCCL = 15
        RCISK = 3
        BCISK = 4
        NRCW = 50
        NRCISK = 2
        READ (RCISK,10) ZTIME, NRED
        FCFORMAT (F6.C,113)
        DC 20 I=1,NROW
        DO 15 J=1,NCOL
            RCATA(I,J) = 0
            BDATA(I,J) = 0
        CONTINUE
15      CCNTINUE
2C      DC 60 I=1,NRED
        DC READ (RDISK,40) (RDATA(I,J) , J = 1,NCOL)
        FCFORMAT (15I4)
40      CCNTINUE
60      READ (BCISK,77) ZTIME, NBLUE
73      FCFORMAT (F6.C,113)
77      DC 83 I =1,NBLUE
        DC REAC (BDISK,81) (BDATA (I,J) ,J=1,NCOL)
        FCFORMAT (15I4)
81      CCNTINUE
83      REWIND NDISK
85      WRITE (NDISK, 90) ZTIME,NRED,NBLUE
        FCFORMAT (F6.C,213)
5C      DC 92 I=1,NRED
        WRITE (NDISK, 91) (RDATA(I,J),J=1,NCOL)
        FCFORMAT (15I4)
51      CCNTINUE
52      DC 93 I=1,NBLUE
        WRITE (NDISK, 91) (BDATA(I,J), J=1,NCOL)
53      CCNTINUE
        REWIND BDISK
        REWIND RDISK
955      STOP
        END

```

INDO00,0
 INDO00020
 INDO00C30
 INDO00C40
 INDO00C50
 INDO00C60
 INDO00C70
 INDO00C80
 INDO00C90
 INDO0C100
 INDO0C110
 INDO00120
 INDO00130
 INDO00140
 INDO00150
 INDO00160
 INDO00170
 INDO00180
 INDO00190
 INDO0C200
 INDO00210
 INDO00220
 INDO00230
 INDO00240
 INDO00250
 INDO00260
 INDO00270
 INDO00280
 INDO0C290
 INDO00300
 INDO00310
 INDO00320
 INDO00330
 INDO0C340
 INDO00350
 INDO00360
 INDO0C370
 INDO00380
 INDO0C390

BIBLIOGRAPHY

1. Center for Advanced Research, SEATAG: A Sea Control Tactical Analysis Game, 2d ed., Naval War College, 1978.
2. Couhat, J. L., Combat Fleets of the World 1978/1979: Their Ships, Aircraft, and Armament, United States Naval Institute, 1978.
3. Naval Postgraduate School Report 55LW73061A, Random Number Generator Package LLRANDOM, by P. A. W. Lewis and G. P. Learmouth, June 1973.

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 55 Department of Operations Research Naval Postgraduate School Monterey, California 93940	10
4. Professor A. F. Andrus, Code 55As Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
5. LT Ellen F. Roland, USN, Code 55Ro Department of Operations Research Naval Postgraduate School Monterey, California 93940	1
6. LT Kevin J. Kelley, USN 211 Threadneedle San Antonio, Texas 78227	1
7. Professor J. K. Hartman, Code 55Hh Department of Operations Research Naval Postgraduate School Monterey, California 93940	1

Thesis
K27143
c.2

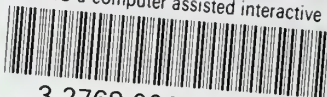
Kelley

CASTAG a computer
assisted interactive
naval wargame.

187921

thesK27143

CASTAG a computer assisted interactive n



3 2768 001 02890 5

DUDLEY KNOX LIBRARY C. 1